Table of Contents

List of Annexes

Revision Directory

Page	Rev.-No.	Date	Revision History
all	0	27.11.2023	New Directive

Preamble

Software is a central component of academic research and the scientific infrastructure and is developed and used in all HZDR institutes. In this regulation, software refers to all forms of program code (e.g. source code together with associated documentation) and executable programs generated from it, which are developed, made available and passed on within the scope of activities at the HZDR. The development of software is part of a creative process and in this sense generates executable knowledge. It is an integral part of modern publication contexts consisting of written publications, data sets and software. In addition, software development is an intellectual and copyright-protected achievement and, in the context of research and transfer, an independent product of scientific work.

Most researchers who write software are not trained software developers and often base their software engineering on their direct working environment and the experience of their colleagues, without necessarily being familiar with the advisory services, tools and good practice experience of the institution. In order to support these researchers in their independence and ability to act, this regulation provides a framework for the development, management and transfer of software at the HZDR. The regulation sets out the basic requirements for the development, management and transfer of software, describes practical processes as well as corresponding examples and advisory services.

The regulation covers the software life cycle, from software development and documentation to the transfer and maintenance of the software. The regulation is intended to support the establishment of modern software engineering methods at the HZDR, which enable high standards in software development, software quality and management. This professionalization will achieve greater sustainability and promote good scientific practice in terms of the verifiability and reproducibility of research results.

It is primarily aimed at all developers and managers who are directly or indirectly involved in the development, management and distribution of software at the HZDR. The aim of the regulation is to establish a confident and sustainable approach to programmed software at the HZDR, to improve software quality and to increase the appreciation of high-quality software.

The regulation takes into account the Helmholtz Association's view of Open Science and is based on the model guideline "Sustainable Research Software at the Helmholtz Centres" [1]. It reflects and implements the regulations for "Safeguarding good scientific practice" [2], the internal requirements of the "Publication Directive" [3], the "Data Policy" [4] and the HZDR Strategy 2030+ [5] in this context. In addition, the Helmholtz "Checklist to support Helmholtz Centres in implementing guidelines for sustainable research software" [6] and the software guideline of Forschungszentrum Jülich GmbH were taken into account in the preparation of this document.

In the spirit of Open Science, access to and reuse of software are the basis for the traceability, verifiability and reproducibility of scientific results. In parallel to the FAIR principles (Findable, Accessible, Interoperable, Reusable) [7], which apply to research data, the HZDR also implements these sustainability principles to software. To fulfil the criterion of sustainability in science, the HZDR encourages researchers to make software openly accessible as "open source". However, at the beginning of a software project, the developing institute must consider which transfer route will have the best impact on society and the economy. If there are commercial applications for the software, commercial exploitation via proprietary licenses may make more sense than open source publication. Proprietary licenses can also enable innovations, e.g. in a company, and secure a technological

advantage, e.g. in a spin-off. Open science on the one hand and commercial utilisation on the other are not contradictory, but complementary elements of the HZDR's innovation strategy.

If required, the advisory services of the Central Department of Information Services and Computing (FWC), the Executive Board Office (FKVB) and the Technology Transfer Department (FSTT) can be utilised.

1 Scope of Application

This directive for the development of research software is aimed at all employees, doctoral candidates, students and guests of the Helmholtz-Zentrum Dresden-Rossendorf e. V. who develop, modify or publish software for the HZDR as part of their duties. The guidelines laid down here regulate the handling, authorizations, legal aspects (copyright, licensing, and exploitation) and infrastructural conditions for the creation and publication of computer programs.

Software is often developed and utilised together with partners in cooperation projects. As this regulation does not apply to external cooperation partners, cooperation agreements with corresponding contribution agreements and stipulations on licensing and exploitation are necessary for the collaboration.

2 Principles for the Development, Documentation and Transfer of Software at the HZDR

Software and software projects differ from one another in terms of both their complexity and their degree of maturity. The complexity of a software project can result from technical aspects but also from management aspects, such as joint software development in a large consortium. The scientific specialism, the programming technologies used, the target group of the project and the potential risk (e.g. liability in the event of malfunctions or failure) that may be associated with a subsequent software product also play a role. In practice, the development of software is an iterative and ongoing activity that changes throughout the software life cycle.

The measures and quality standards that the HZDR defines for software quality are based on the application class of the software developed. At the beginning of a project, those responsible for each software development define an application class for which the described quality and documentation measures are to be aimed for. The application classes make it possible to organize activities and the use of tools according to requirements. The recommendations are to be interpreted and evaluated in the context of the respective project. If the desired application class is not achieved, the person responsible for the software defines suitable measures for further development together with other technical stakeholders. In this context, the cost-benefit ratio must be considered realistically in view of the remaining development time and resources. As experience shows that many projects evolve, it is advisable to use standards of higher application classes right from the start. This classification is based on the application classes proposed and implemented by the German Aerospace Centre (DLR) [8]. The quality requirements for software engineering increase with each application class.

2.1 Classification according to Application Classes

The application classes are primarily based on how the software is used and whether or to what extent it is to be passed on. A planned or required transfer of the software is a decisive criterion for the selection of the application class [8].

Table 1: Brief overview of the application classes for software [1)]

Application Class	**0**	**1**	**2**	**3**
Development	at the HZDR	at the HZDR	at the HZDR or with third parties	at the HZDR or with third parties
Utilisation	Personally and internally in the project team	at the HZDR	Planned as a long-term usable result from third-party funded projects or as a product.	Product character or software for services (e.g. cloud platform)
Requirement	Compliance with legal aspects, version control system is used	Version control systems and "good software practice" are used. This makes further development possible for those not involved in the HZDR.	Maintainability and usability are given. Rights of use and exploitation are held by the HZDR.	Test automation, release and maintenance management are provided
Transfer planned?	not planned outside the development team	only within a narrowly defined framework, e.g. the institute or the supervisor's institution	yes, redistribution with license (OSS or proprietary)	yes, redistribution with license (OSS or proprietary)
Examples	Code on a small scale, individual functions, simple scripts	Software as a result of research work (e.g. dissertations) with a demonstration character	*Software publications*, software developed in co-operation with partners	Software that is to be commercially exploited or software that is created in a large open source project

The application classes are primarily based on how the software is used and whether or to what extent it is to be passed on. A planned or required transfer of the software is a decisive criterion for the selection of the application class.

However, the measures and standards for the corresponding application class relate primarily to software development and documentation. The complexity of a software project and the maturity of the software are therefore not necessarily reflected in the application classes. It is possible to switch between the application classes, but usually from lower to higher classes. It can therefore be helpful

1) A detailed description and explanation of the criteria that lead to the selection of the application class can be found in [8].

to orientate yourself to the standards of the higher application class if a transfer of the software is not ruled out. The necessary use of resources must be taken into account, which must be ensured over the entire software life cycle at HZDR.

2.2 Minimum standards and measures for the application classes

Application class 0:

1. Observing legal aspects (e.g. third-party rights) is an integral part of the software development process and requires appropriate documentation (e.g. copyright notices). The checklist (Appendix 1) provides assistance on this topic.

> Example: If programming is based on existing software or libraries are used, these license conditions must be observed.

A version control system must be used from the outset so that all contributions to the software are documented.

From application class 1 additionally:

2. Version control systems and code repositories are used that contain or reference all components of the software required for use and are used for change management.

> Example: Any changes to the source code are documented using Git.

3. A short, meaningful description of the purpose of the software must be included (e.g. in the form of a README file in the repository). New contributors should be able to participate in the development of the software as documented in the README file or a contributions file.

> Example: Explanation of the scientific context of the software, reference to publications and participants as well as technical information on use and further development. This makes access easier for new contributors in particular.

4. To ensure citation, machine-readable metadata must be included as a citation reference [9].

> Example: The "Citation File Format" and "CodeMeta" allow indexing and displaying of citation references on platforms such as Github.

5. The tools and standards for software development, validation and verification must be agreed with the development team (and superiors) and their use must be documented. Where possible, software development is carried out according to recognised standards and with state-of-the-art tools during development, validation, verification and deployment.

> Example: Tools such as SoftWipe https://github.com/adrianzap/softwipe and publications such as "Best Practice for Scientific Computing" (Wilson et al. 2014; PLOS Vol.12 Issue 1). More good practice examples, tools and publications are listed on the intranet portal (see chapter 3).

6. Versions (releases) should be clearly labelled for users, for example using release numbers. For large software projects, it is advisable to define a joint re-release policy.

> Example: Git "tags" are enormously helpful for reproducibility of results and communication, especially in combination with a persistent identifier.

7. For the purpose of referencing and citing the software, versions (and possibly the software as such) must be assigned a persistent identifier (PID). The necessary metadata can be found in the citation note (see above).

> Example: Software that is not publicly accessible can also be FAIR and receive a DOI, for example, which is easier and more established for use in text publications. In addition, a DOI enables subsequent relocation without breaking the reference. Reproducibility is greatly facilitated by referenceable versions of data, software and environment.

8. Every software publication of the research centre must be documented with at least metadata in the same way as data publications in the institutional repository. If no publication is produced, software should be made findable in a central internal software catalogue.

From application class 2 additionally:

9. Appropriate installation, development and application documentation is provided for the distribution and use of the software. The documentation should be easy for others to understand in order to facilitate contributions from other contributors.

10. Functional tests of the software are carried out systematically. For this purpose, an adequate test strategy is defined (e.g. sanity and regression tests) and documented.

11. If the software is developed as part of a co-operation, the grant conditions and the provisions of the contracts or the grant agreement and/or the specifications of the consortium must be observed.

If the software is developed as part of a co-operation, the grant conditions and the provisions of the contracts or the grant agreement and/or the specifications of the consortium must be observed.

12. Before the software is passed on to third parties outside the research centre, a license is defined for the software that regulates the use of the software by third parties. License terms for third-party software used are complied with. The software will not be passed on without a license. A copyright notice is required for the transfer.

> Example 1: The software is provided as open source. A suitable open source license is selected by the institute with the help of the checklist and used for distribution (see 3.3).
>
> Example 2: The software is passed on with restrictions. The Executive Board Office / Legal Affairs (FSVB) draws up an individual software license agreement together with the institute, which regulates the use of the software.

13. The approval processes of the research centre / institute are observed for the transfer of software.

> Example 1: The software is to be made open source. The respective institute management or a person authorised or appointed by the institute management approves the publication.
>
> Example 2: The software is to be passed on to third parties to a limited extent. The institute consults with FSVB. Together with the institute, FSVB draws up an individual software license agreement that regulates the use of the software.

14. Before transferring to third parties outside the research centre, it must be ensured that the export control regulations (foreign trade law) have been complied with.

Example: Before publication, the Institute uses the EC Dual-Use List to check whether the software is listed. If there are any indications of this, the Legal Division (FSVB) must be involved. For example, software is listed in accordance with 3D001 if it has been specially developed for the development or manufacture of equipment for the production of semiconductor components or materials. Instructions for checking against the list can be found on the intranet portal.

From application class 3 additionally:

15. A high degree of coverage with test automation is provided.

    Example: Continuous integration or continuous delivery can be used for test automation. GitLab CI, GitHub Actions etc. are easy to use and well integrated into the web interfaces. The tests are automated with tool support (e.g. with pytest for Python code).

16. The Institute has organised software maintenance and troubleshooting.

    Example: The software can either be maintained with your own resources or the community can take over maintenance and troubleshooting. It is also possible to commission a company with this task.

Exceptions:

So-called snippets, short excerpts from the source code, do not fall under any application class and can be distributed without a license.

2.3 Provision, transfer and citation

Components for the sustainable provision of research software are archiving, the use of appropriate infrastructures and medium and long-term maintenance.

The HZDR welcomes and supports the sharing of software developed at the HZDR, regardless of whether the software is shared in source code or object code, and regardless of whether the code is shared with a proprietary license, i.e. for restricted use, or open source. When publishing as open source software, the reach and transfer effect is enormous, as millions of developers are active on platforms such as GitHub. For this reason, and to fulfil the criterion of sustainability in the scientific community, the HZDR encourages researchers to make software openly accessible as "open source" (see Chapter 3.3).

According to the above-mentioned categorisation of software into application classes, software in application classes 2 and 3 is suitable for redistribution. For each redistribution, even if this is done as a freely accessible publication, e.g. via repositories, journals and archives, usage rights to the software must be defined within the framework of a license. It must be carefully checked whether the license terms of any third-party software used are complied with.

For each provision or publication of new software by HZDR employees, the approval process in the HZDR publication system (ROBIS) must be completed. The application class is determined by the respective institute on its own responsibility and updated in the event of changes. Responsibility can be delegated by the institute management to the project managers or software developers. A new release is only necessary if the application class or the contractual framework conditions for a software change, not if it is republished due to release or functional changes. Responsibility can be

delegated by the institute management to the project managers or software developers. A new release is only necessary if the application class or the contractual framework conditions for a software change, not in the event of a new publication due to release or functional changes.

> An exemplary release process for the initial release of class 3 software (analogous to publication) is: Software developer → Project manager/group leader → Entry in the publication database → Release by department/institute management with automatic information to FSTT → Publication of the software, e.g. via GitLab / GitHub / RODARE

The checklist (Appendix 1) must be completed and the decision tree (Appendix 2) used to implement the measures for the provision and transfer of software. The completed checklist is filed as a document in the approval process.

The following applies to the citation of software:

I In accordance with the DFG's Guidelines for Good Scientific Practice, the HZDR requires its employees to acknowledge the use of their own or third-party scientific software by citing or referencing it in their publications.

Citation and referencing are the core elements for the reproducibility of results and the visibility of non-scientific employees. Ideally, FAIR software publications with persistent identifiers are available for this purpose. The necessary data can usually be taken from these. If these are not available, citation information may be available in the code repository.

Regardless of its public availability, software must be cited in a scientific publication with the following information [10]:

> Authors, name of the software, PID, release date, [possibly existing release number]

If there is no PID, a revision and the URL of the source code repository (or a qualified software archive such as the Helmholtz Research Software Directory or the "Software Heritage Archive" [11]) can be specified as an alternative.

The following applies to the provision and transfer of software:

1. The software was created in accordance with the standards of the respective application class (see section 2.2) and is documented and versioned accordingly.

2. The (copyright) legal aspects, potential third-party rights, funding requirements and release processes must be observed for the transfer and publication of the software.

3.The desired transfer path (open source/proprietary license) should be defined before software is made available and passed on to third parties.

> Important points to consider here include the degree of openness and the access granted, possible commercial utilisation plans and any (legal) restrictions. This forms the basis for the decision on the transfer route. If necessary, the counselling services and decision-making aids mentioned in chapter 3.1 can be consulted for this purpose.

4. Whenever possible, developers shall make the software on which the publication is based accessible in recognised archives and repositories.)

5. Irrespective of the accessibility of the software itself, metadata on the software must be published in the sense of a software publication (see 2.2).

6. Irrespective of the type of provision and distribution, whether freely accessible or proprietary, the user must be granted the corresponding rights of use by means of a license (see chapter 3.3).

7. The citation should clearly identify the SW via PID (e.g. DOI), with a reference to a separate document (contributor list) in which all authors are named. The authors can also be named in the source code but without the copyright designation or symbol.

8. The authors of software are encouraged to promote the traceability of the use of software.

9. Quality assurance and software publications are included in the scientific reporting to reward the performance in the creation, maintenance and care of the research software.

10. The realisation of long-term accessibility should be ensured in compliance with quality standards.

11. The publication can be linked to embargo periods (analogous to retention and embargo periods for data and text publications).

3 Support and services at the HZDR

In order to support the development of software that meets high quality standards at the HZDR, the research centre provides consulting, support and training services. Advice and information relate to all phases of the software life cycle and the dissemination of software.

The various offers on the subject of software are documented in detail centrally on the intranet portal and updated regularly. Additional information such as good practice examples, FAQs, contact persons and announcements on training and further education can also be found there. The portal is designed jointly by the Executive Board Office, the Central Information Services and Computing Department, the Technology Transfer and Innovation Department and the software developers.

3.1 Counselling services and decision-making aids

The information and tools provided on the intranet portal offer practical help. The checklist (Appendix 1) and the decision tree (Appendix 2) help to clarify open points and important aspects of the decision-making process before passing on software.

Supplementary information such as tutorial collections, good practice examples and answers to frequently asked questions are also made available via the intranet portal. These can be used to review or assess your own software project and as preparation for personal consultations. In addition, the good practice examples mentioned at the HZDR can be used as a decision-making aid for common software procedures and common open source licenses and their legal classification.

If you have any questions or uncertainties regarding the use of self-programmed software, you can arrange a personal consultation with the contact persons at the FWC, FSTT and/or FSVB, whose current contact details can be found on the intranet portal. During the consultation, questions such as what effect the software should have, which transfer or publication channel is suitable (e.g. open source software or proprietary software) and which (copyright) legal framework conditions are given or must be taken into account are clarified. If proprietary licensing of the software is desired (see section 3.5), the framework conditions are clarified in a consultation and a suitable license agreement is then drawn up.

3.2 Support and further training programmes

Training courses on software engineering play an important role in enabling young scientists to develop good software and to implement the standards for good software at the research centre. The HZDR would therefore like to create easy access to internal and external training programmes. Furthermore, the existing range of training courses on software development methods and good practices at the research centre is to be further expanded in the future.

The intranet portal lists internal and external offers and information, which are regularly updated and continuously supplemented. The Helmholtz Association's infrastructure and activities such as the Helmholtz Codebase, the HIFIS Software Helpdesk, HIFIS and HIDA courses and training programmes and the Helmholtz Open Science Forum "Research Software" can all be found there.

The HZDR encourages all software developers to actively utilise the aforementioned offers and to get involved themselves. Managers are called upon to encourage suitable employees to become involved as trainers in the field of software development (e.g. as part of carpentries workshops or internal training programmes) and grant them the necessary freedom to do so.

The establishment of a network for developers of research software is actively promoted in order to create a forum for community building and exchange of experience (e.g. through mailing and chat lists, forums, networking events). This includes supporting the collaboration of developers in Helmholtz-wide activities and open source software and the expansion of cooperation with other university or non-university institutions in the field of software development.

3.3 Scientific and commercial exploitation and choice of license

The HZDR supports and advocates the publication of software as open source software in order to contribute to a strengthening of "open science" and thus to achieve a more effective and open exchange of information within science and to promote the transfer of results to society.

In cases where commercial utilisation of the software is possible and sensible and is in the interests of the software developers and the institute, the HZDR recommends such commercial utilisation. The scientific use and commercial exploitation aspects can certainly be joint components of a coordinated exploitation strategy for software and thus best protect the interests of the institute and the HZDR for the creation, transfer and provision of software.

To decide on a specific license type, the application class must be determined early on in the software development process and in consultation between software developers, managers and project managers.

- to determine the application class,
- to ensure the quality of the software,
- determine the target user group,
- consider the method of distribution (license type) and
- examine the legal framework.

A checklist and a decision tree are available via the intranet portal to help with these decisions.

Only if the HZDR has all the necessary usage and exploitation rights to the software can it be passed on in compliance with any applicable (open source) license conditions.

This applies in particular to derived works. The choice of license is the responsibility of the respective institute and can be delegated to the project managers, executives and software developers as required and depending on the process. If there are any questions or uncertainties regarding the transfer of software or the choice of license, software developers and project participants can contact the contact persons on the developer portal in order to jointly analyse the respective situation and find the best possible solution.

Depending on the contractual framework conditions (cooperation agreements, consortium agreements, funding regulations, etc.), the third-party software that may be included and connected, the desired transfer path, the applicable application class, etc., different licenses can be selected for the transfer of the software. There is a choice between various open source licenses [12,13] or proprietary licensing.

3.3.1 Open Source Licenses

When publishing software as open source [12,13], the HZDR recommends a license issued by a recognized license of the Open Source Initiative. Open source licenses are recommended for HZDR, which can be selected without explicit advice from the legal department. A decision aid for selecting one of these licenses is provided in Appendix 2.

License texts and current information on the recommended open source licenses as well as tools for compatibility checks are linked in the intranet portal. If a deviation from the above-mentioned licenses or advice on the selection of a suitable license is required, the Legal Department (FSVB) or Technology Transfer and Innovation (FSTT) must be consulted (see section 3.1).

3.3.2 Proprietary Licenses

If the aim is to exploit the software commercially or to create a more individual license, e.g. because the distribution of the code or the type of use of the software (e.g. commercial use) is to be restricted, a proprietary license is also an option. Contractual models are also possible in which the software is licensed proprietarily but free of charge for academic purposes, but commercial use is subject to a fee (so-called dual licensing) [13].

If there is a need to freely use and modify software only within a closed community, proprietary licenses are a good option, as an existing open source license conflicts with a closed community. In the case of joint software development in large communities whose parties also contribute background software to the collaboration, it is essential to pay attention to license compatibility.

Software that is provided with an open source license can also be commercially exploited if the HZDR holds the exploitation rights. Various models can be pursued for the utilisation of open source software. If software is used as background IP in collaborative projects, it must be provided with a license before being passed on to partners, as software may not be passed on without granting explicit rights of use (open source or proprietary), even in collaborative projects.

In the event of uncertainties, questions arising or as a decision-making aid for the various transfer paths, the above-mentioned counselling services available via the intranet portal can be used for support. If necessary, it can be checked together in each individual case whether, for example, the rights of third parties are being infringed and whether this would restrict or even prevent scientific or commercial use or subsequent utilisation.

Technology Transfer and Innovation (FSTT) at the HZDR is responsible for negotiating any license agreements that grant proprietary use. If proprietary licensing is desired, advice is given together with the institute on possible license conditions, contractual framework conditions, the intellectual property situation, etc. and a corresponding license agreement is drawn up. FSTT consults with FSVB on the contractual framework conditions and negotiates the license agreement with the potential licensee.

4 Conclusion

Research software is an important infrastructure and a valuable result of scientific work. The implementation of this software guideline at the HZDR enables employees to develop and pass on software sustainably and with high quality standards within a practical and reliable framework.

The division into application classes, the identification of quality standards and measures, the consideration of possible transfer paths or license types and the associated legal aspects ensure that professionalisation is achieved across the board at the centre.

This will be supplemented by decision-making aids and advisory services from Legal, Information Services and Technology Transfer provided by the HZDR and channelled via the intranet portal in the future. In addition, the HZDR would like to further expand existing internal training programmes on software development methodologies and best practices in the future in order to enable young scientists to achieve high standards in software development and documentation.

The impact generated by this scheme is intended to create sustainability and good scientific practice in the development and dissemination of software and thus create added value in society, business, science and politics.

5 Coming into force

This instruction comes into force upon signature by the Executive Board and replaces all previous regulations on the subject.

6 Further information

[1] Model Guideline for Sustainable Research Software at the Helmholtz Centres, doi.org/10.2312/os.helmholtz.007

[2] HZDR Regulation B110: Rules for Ensuring Good Scientific Practice

[3] HZDR Regulation B213: Publications Directive

[4] HZDR Regulation B220: Data Policy

[5] HZDR Strategy 2030+ - Moving Research to the next Level for the next Gens

[6] Checklist to support the implementation of guidelines for sustainable research software, doi.org/10.48440/os.helmholtz.031

[7] The FAIR Data Principles, force11.org/info/the-fair-data-principles

[8] Software engineering recommendations of the DLR, doi.org/10.5281/zenodo.1344608

[9] How to cite software: current best practice, doi.org/10.5281/zenodo.2842910

[10] FORCE11 Software citation principles, doi.org/10.7717/peerj-cs.86

[11] http://www.softwareheritage.org/

[12] http://www.opensource.org/licenses

[13] Example 13CFLUX2, 13cflux.net/13cflux2/

[14] Use of Open Source Software at the DLR, www.dlr.de/de/medien/publikationen/broschueren/opensource-software_dlr_2022.pdf

[15] Institute for Legal Issues of Free and Open Source Software, ifross.org

[16] Semantic Versioning, semver.org

[17] Research Software Engineers (RSEs), de-rse.org

[18] BMJV Rechtssetzung Bürokratieabbau

7 Explanation of terms

Derived works:

Rights must also be granted by the author or their employer when using or integrating code whose rights of use are not (exclusively) held by the HZDR, but by third parties, e.g. cooperation partners (third-party software). The license conditions of the (open source) software used must also be observed, as non-compliance constitutes a breach of contract and can lead to claims for damages. This also applies if only programme components or partial sequences are integrated into your own software. In this case, an intensive examination is required to determine the extent to which the various licenses of the third-party software to be integrated contain incompatible regulations. This can lead to the own software not being able to be published under the intended open source license. In addition, the open source license of the software used may expire if the license conditions are not complied with.

Application class:

Application classes [8] are used to categorise software and define associated rules and recommendations with regard to appropriate software development practice and documentation. They make it easier to check the rules and communicate on the associated topics. Further information on the term can be found in the document "Software Engineering Recommendations of the DLR" (from page 7) [8,15]. It contains an exemplary definition of application classes that build on each other. The described "application class 1", for example, defines minimum recommendations for small, non-critical software (e.g. data evaluation scripts) and is compatible with the minimum practice defined in this document with regard to development and documentation.

Archiving:

Archiving the software programmed at the HZDR contributes to the sustainability of software development. Archiving primarily serves to secure versions for verification, provenance and reproducibility purposes in the context of scientific results. Archiving includes accompanying data (e.g. metadata, documentation, runtime environment and test data if applicable). In addition, storage periods must be adhered to in accordance with the rules of good scientific practice in the specialist disciplines.

Depending on the requirements and application class, archiving options may include: repositories (RODARE, Zenodo etc.), software journals and the Software Heritage Archive. Further information can be found in the Best Practices and on the intranet portal.

Sustainable software requires that even inactive or archived software remains usable. The fast pace of technology in the software environment poses a challenge when it comes to archiving software. When planning, implementing and utilising self-programmed software, software developers, project managers and executives must decide together how to act in each specific case based on the application class. Those responsible for the project must plan the appropriate (long-term) resources for this from the outset.

Contributor Agreement:

A Contributor Agreement or Contributor License Agreement (CLA), is a document describing the conditions under which intellectual property can be contributed to a project or endeavour; usually a software project under an open source license. The HZDR provides a verified template via the intranet portal, which can be submitted to external contributors.

Developers

Developers are people who develop software as part of their job. This includes both scientists who develop software and software developers who develop software in scientific projects.

Export control:

As soon as persons from countries outside Germany / the EU are to be involved in the project, a foreign trade law review of the project may be necessary. A checklist on the intranet portal can be used to determine whether this is necessary for your own project.

Git:

Git is a version control system that can be used to track who made which changes to the code and when. Git offers options for easily updating a released or public version of a code on GitHub.

Information security:

The IT security officer provides a classification system for the level of data protection in the "Information Security Guideline" (in preparation). When creating software, it is therefore recommended to consider the categorisation of the incorporated data and the software itself based on the guideline at an early stage. A classification system must be in place at the latest when sub-substantial parts of the code are available.

Licenses:

OSS licenses: The general view is that the Open Source Definition correctly reflects what is understood by the term "open source software" in the open source community. The decisive criterion of the definition is that the license of a software allows the comprehensive "free" use of the program and does not restrict the licensees or areas of use, i.e. it may be used by anyone for any purpose. The absence of license fees and the openly accessible source code are essential prerequisites for free usability. The license agreement under which the software is offered is therefore always decisive [15].

Proprietary licenses: The main difference between proprietary and open source licenses is that the proprietary license can be designed completely freely and, in particular, opens up the possibility of completely restricting the distribution of code. Although the use of an open source license does not impose a blanket obligation to redistribute, this right cannot be restricted vis-à-vis third parties. Furthermore, the open source license only applies if the third party redistributes the software. For example, it would be possible to offer a service with the software without passing it on. A proprietary license therefore becomes interesting if you want to restrict the passing on of the code itself or a certain type of use (commercial use).

Copyleft licenses: Further developments of the software must be distributed under the same license conditions as the software itself.

- Strict copyleft: no exceptions! (GPL, AGPL, CPL) All adaptations must be subject to the original license.
- Limited copyleft: exceptions are possible (MPL, LGPL)-exceptions to the strict copyleft are permitted.

Non-Copyleft licenses: Further developments of the software can be released under different license conditions than the software itself (e.g. BSD, Apache)

- No obligations for the licensing of new or modified code
- Licenses with special rights - (e.g. NPL, QPL). The processor must grant special rights to the owner of the original software.
- Licenses with optional rights (Artistic/Perl). The processor of a software may choose between various options for licensing his/her processing.

Right of use and utilisation:

As soon as software developers are in an employment and service relationship with the HZDR, the rights of use and exploitation automatically belong to the research centre in accordance with copyright law. Special arrangements must therefore be made before the start of the project if third parties are to be involved who do not have an employment contract or similar with the research centre. External contributions can be integrated via a Contributor License Agreement if required. The HZDR provides a verified template via the intranet portal, which can be submitted to external contributors.

PID:

Persistent Identifier

Project manager:

In the context of scientific projects, a manager is usually entrusted with the management of the project. This person may also be responsible for the software, depending on the size and share of the research software in the project.

Software Publication:

Analogous to research data and scientific texts, software is also a result of scientific work, so that it can be made known to a wide audience as a scientific publication. Nevertheless, access to the software from the publication may be restricted in the same way as for research data. Like research data and text publications, a software publication contains metadata with which it can be found and cited.

Release:

A release is a version of software that is made available to users or contributes to a scientific publication. Following a scheme [16], a release number ensures that the release and the associated content are clearly labelled.

RSE:

Research Software Engineer [18]

Software:

In the context of this regulation, software refers to all forms of programme code (e.g. source code together with associated documentation, parameters and workflows) and executable programmes generated from it that are developed and/or (re)used as part of an activity at HZDR.

Research software includes software that is developed as a result of research work, is used as a solution in research or serves as an infrastructure component for research work. All three categories are developed at the HZDR, so this policy covers all of them.

Software life cycle:

The life cycle of software describes all key development stages, from the idea and concept, through development, use and maintenance, to archiving and decommissioning.

Standards / current state of the art:

The term "state of the art" is a legal-technical term that implies certain minimum requirements for the product. Specifically, it means: The state of the art is the state of development of advanced processes, equipment and operating methods which, according to the prevailing opinion of leading experts, appears to ensure the achievement of the legally prescribed objective. Procedures, equipment and operating methods or comparable procedures, equipment and operating methods must have proven themselves in practice or - if this is not yet the case - should have been successfully tested in operation.

The position of the term becomes clearer when compared to "generally recognised rules of technology" (below) and the state of the art in science and technology (above) [18]. (see 4.5.1.)

There are basically two possibilities. Either you define your own criteria that are as specific as possible, or you decide in favour of one of the three terms and ensure that they are used correctly within the project.

Copyright (Urheberrecht)

Software is subject to copyright. § Section 69a of the German Copyright Act (Urheberrecht, UrhG) protects software in any form, including design material, in all forms of expression (QC, C, EXE, modules). However, ideas and principles on which the work is based are not protected.

Although moral rights are not transferable under German law, the granting of extensive rights of use and agreements on exploitation rights are permitted. If, for example, American law is taken as a basis, then an assignment of the entire copyright is also possible.

As soon as the software developers are in an employment and service relationship with the HZDR, the rights of use and exploitation automatically lie with the research centre (details are regulated in Section 69 b of the German Copyright Act). Special regulations must therefore be made before the start of the project, for example in the case of spin-offs or commercial use, if third parties who do not have an employment contract or similar with the research centre are to be included in the plan.

Version control system:

The use of version control systems is important in the development of high-quality software in order to maintain an overview of all changes to the code and the software developers involved. These should be linked to collaborative functions for projects and communities. The version control system is used to record changes to data (e.g. documents, programme code) by a defined group of people, which are managed in a shared storage location ("repository"). Each change is saved with a time stamp and the author of the change. The resulting history makes it possible to retrace changes and return to previous statuses.

The HZDR currently offers such a collaborative platform for its employees and partners in the Helmholtz Association: Helmholtz Codebase. In addition, it is desirable that this platform can be used for national and international cooperation by connecting established identity federations such as currently (2021) eduGain, DFN AAI and/or the Helmholtz AAI and making them usable for non-members in this sense.

Other such platforms can also be offered for closed groups or for other important reasons. The global platforms provided or established by the Helmholtz Association as part of HIFIS, such as GitHub.com or GitLab.com, are also available for selection. These can be used at the joint discretion of the developers, managers and project managers, for example to increase reach and visibility.

8 Annex 1: Checklist for the transfer of software

Date:	…
Name of the Software:	…
Institute:	…
Contact:	…

Copyright and rights of third parties

	"All rights of use must be held by the HZDR" "All authors must be known"
☐	All authors of the software are known and named.
☐	All authors have programmed as employees of the HZDR and the rights of use are held by the HZDR. **If no:**
☐	- Third-party institutions are known.
☐	- The HZDR holds the rights of use of these institutions in written form.

Contractual obligations

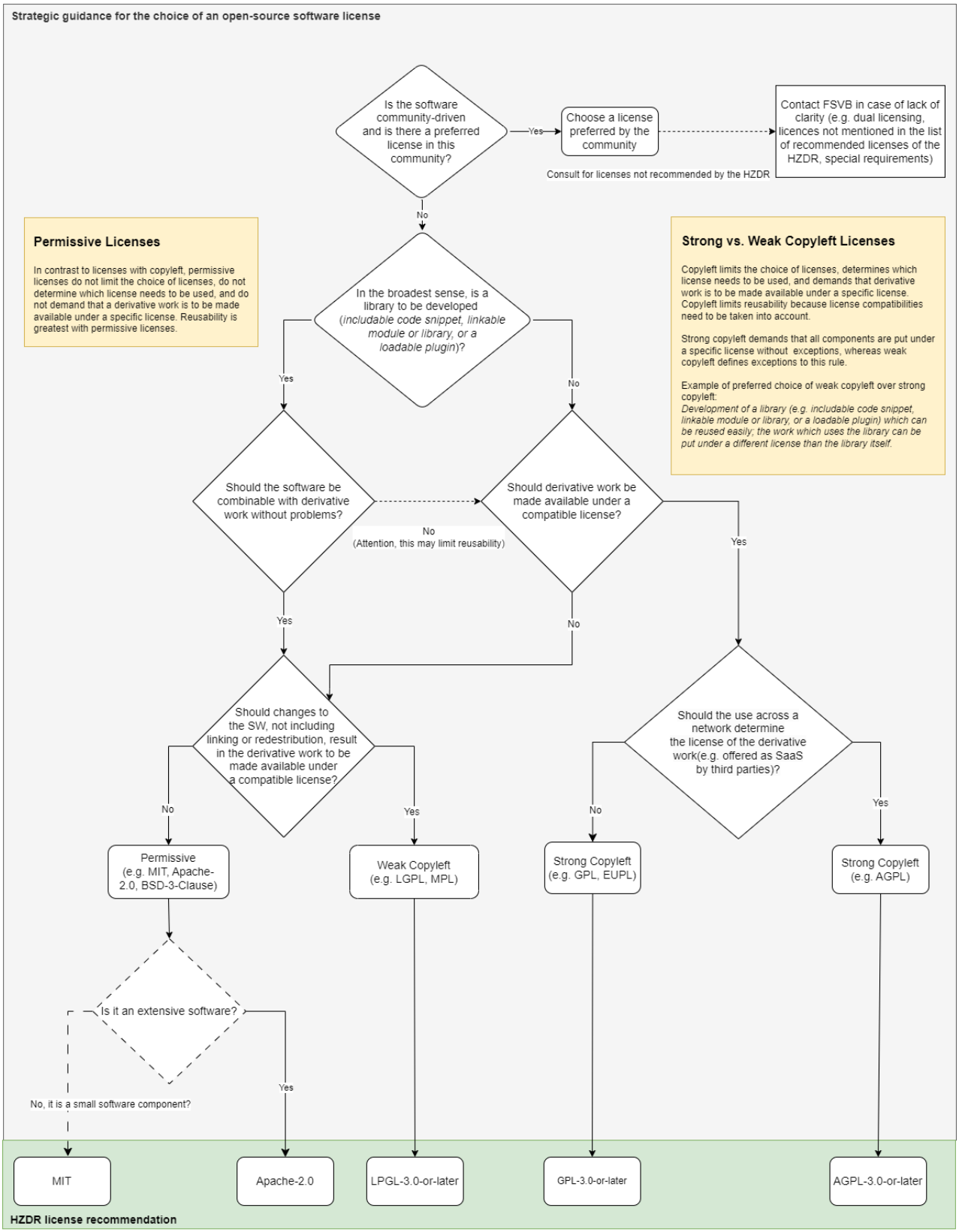
☐	Funding or grant requirements, cooperation agreements, grant agreements and employment contracts were checked for possible requirements or restrictions on the publication and distribution of software.
☐	It is known when/if the software is used as background in projects.
☐	In the case of medical software, the limitations are known and are taken into account.
☐	The export control regulations have been reviewed and are complied with where necessary.

Compatibilities

☐	The software was written without integration of pre-existing software parts or libraries. **If no:**
☐	- The license conditions of the pre-existing/modified software or the linked libraries are known and compatibilities are observed.
☐	- If a paid developer license for the existing software/library is required for the licensing/distribution of your own software, this is available.

If you have any questions, please contact the Information Services and Computing (FWC) and Technology Transfer (FSTT) departments for advice. Further information can be found in the -> Intranet-Portal.

9 Annex 2: Open source licence decision support



It is necessary to check the basic legal conditions, possible questions can be:

- Are the dependencies of the software compatible with the strategic license decision?
- Do all project partners agree with the license decision?
- *In case of lack of clarity get into contact with the legal consulting!*