A Guide for Publishing, Using, and Licensing Research Software in Germany

Alexander Struck[1], *Axel* Loewe[2], *Elke* Achhammer[3], *Fabian* Rack[4], *Felix* Bach[2], *Frank* Löffler[5,6], *Gunnar* Seemann[7,2], *Hartwig* Anzt[2], *Maximilian* Funk[8], *Stefan* Unger[9], *Stephan* Druskat[10,1], and *Sven* Friedl[11]

[1]Humboldt-Universität zu Berlin, Berlin, Germany
[2]Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany
[3]Technische Universität München, München, Germany
[4]FIZ Karlsruhe - Leibniz Institute for Information Infrastructure, Karlsruhe, Germany
[5]Friedrich Schiller University, Jena, Germany
[6]Louisiana State University, Baton Rouge, LA, USA
[7]University Heart Centre Freiburg Bad Krozingen, Freiburg, Germany
[8]Max-Planck-Gesellschaft e.V., München, Germany
[9]Julius Kühn-Institut (JKI), Quedlinburg, Germany
[10]German Aerospace Center (DLR), Berlin, Germany
[11]Berlin Institute of Health, Berlin, Germany

Abstract. Research software has become a central asset in academic research. In Germany, the German Research Foundation (DFG, Deutsche Forschungsgemeinschaft) recently updated the Guidelines for Safeguarding Good Research Practice. Research software is now valued similarly to classic publications and data with implications for research software sustainability and legal aspects.
In this document, we present four decision trees and corresponding legal documentation tables to aid researchers. The decision trees should ease to identify i) the software policy of your institution, ii) restrictions imposed by contributors and the environment, iii) licensing collisions if 3rd party software is included, and iv) problems in licensing (existing) research software.

1 Introduction

In research, many results are based on some kind of software. In some cases, commercial or freely available software is not suitable to fulfill the desired needs and third-party code may introduce additional challenges. In these cases, new software is being implemented by scientists and/or research software engineers (RSE). When starting to do so, it is rare that the developer thinks beyond his or her intended temporary use-case or project. Research is often project-based with strict deadlines and need for scientific output (e.g. Master or PhD but also other third-party funded projects). There is often no time or awareness for anything else. If the software becomes useful for a wider community, indicated by, e.g., recognition in publications or further funding, two main problems arise often: The code was not developed under consideration of sustainability (discussed in our accompanying white paper [1]) and the developer had no knowledge or awareness of legal aspects and thus did not consider them early enough. Additionally, in Germany, the German Research Foundation

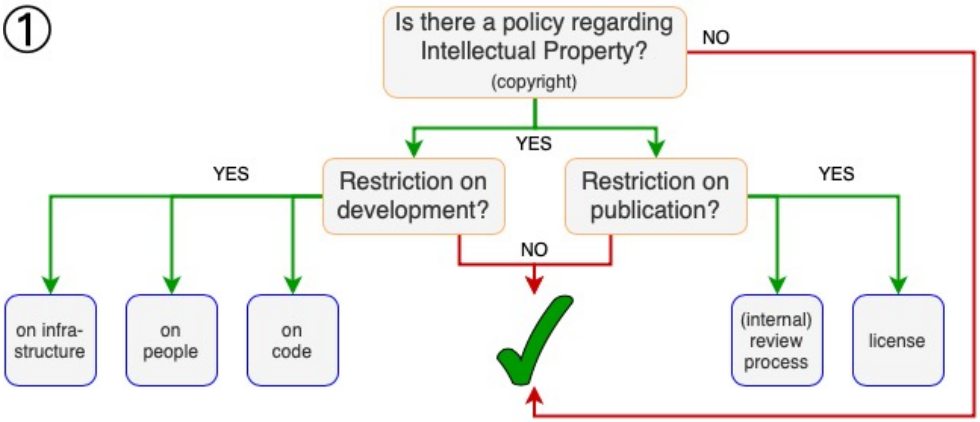

Figure 1. Policy. This tree asks to check closely any policies implemented in the software developers organization. The corresponding documentation can be found in Table 1.

(Deutsche Forschungsgemeinschaft, DFG) recently updated the Guidelines for Safeguarding Good Research Practice [2] including research software as part of the scientific outcome. This may lead to the need to comply with certain standards, e.g. legal aspects.

In this paper, we present decision trees and corresponding documentation tables that can help to clarify and document legal aspects for research software development in Germany. A white paper on research software sustainability was developed in parallel [1].

2 Decision trees

The decision trees presented here shall help developers (in our and similar jurisdictions) to identify risks regarding the mandate of the software. In a perfect world, one would address the legal aspects at the start of a project. It is crucial to know about these to create sustainable software. We strongly recommend writing documentation of the answers and outcomes. The corresponding documentation tables are referenced in each decision tree. Please keep in mind that only restrictions from copyright law are addressed. In some projects, you might also have to consider patents, trademarks etc.

If you need to be entitled to publish, use, and/or license a software you have to check:

- The policy of the institution (Fig. 1)
- The rights restriction imposed by the persons who "create" the software (Fig. 2)
- The rights restriction imposed by the environment (Fig. 2)

Table 1. The policy of the institution. Does your organisation have a policy regarding intellectual property created in the organisation?

	Name	File location	Does it contain restrictions on publication?	Does it contain restrictions on the development (infrastructure, people commiting, coding)
Yes / No			y/n	infra / people / code / none
⋮				

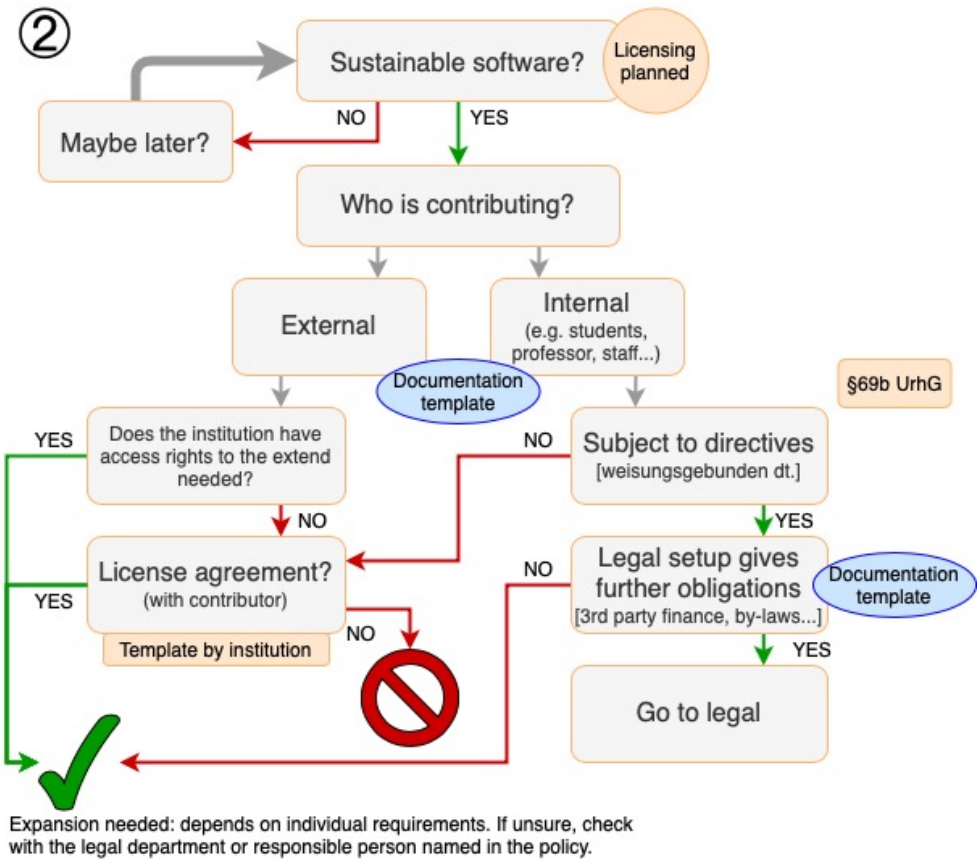

Figure 2. Contributors. This tree helps to find out, if the academic institution where the software development is located is the owner of the intellectual property (copyright). The documentation of contributors can be found in Table 2. The documentation of further obligations is suggested in Table 3.

Table 2. The rights restriction imposed by the persons who "create" the software. Who has or will be contributing to the software development? Entries are exemplary.

Name	Internal (kind of contract, time) OR External (where?)	Code contributed	Status of employment	Other employments (where?)	3rd-party funding (link to Tab. 3)	Signed CLA?
	Int, TV-ÖD, 10/20-09/23		PhD			
	External		Student			
	Int, internship		Postdoc			
	External, guest		Prof.			

- If third-party code is incorporated (Fig. 3)

We also built a tree for the scenario that you have to check on an already existing software (Fig. 4).

Table 3. Rights and restriction imposed by the environment. What is the legal setup and which other obligations does it give?

Is there a third party funding?	Yes/No
Is there a third party involved (cooperation, etc.)?	Yes/No

Funding body	Contact name	File location
⋮		

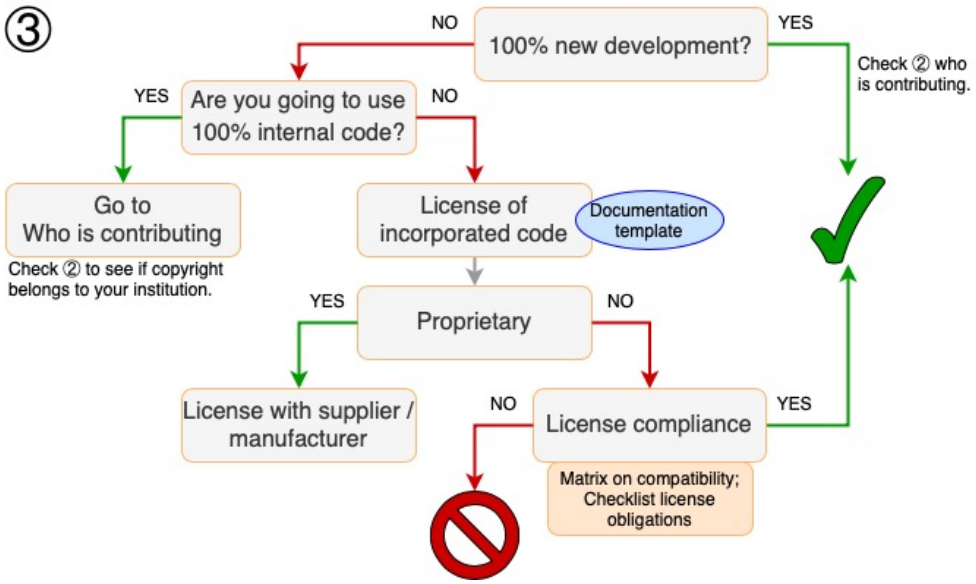

Figure 3. Code history. The code history tree points out tasks for projects that incorporate existing code. The documentation of incorporated code can be found in Table 4

In case of a negative outcome (a red forbidden sign), we believe there is no other solution than to rewrite conflicting code. In a positive outcome (a green check-mark), you have the rights you need to proceed. The other outcomes are self-explaining (e.g. consult your legal department).

Table 4. 3rd-party code. Is third-party code incorporated?

SW name	License type	License name	Purchase date	Allowance to modify code?	How/where incorpo-rated?	download location	Kind of SW
⋮							

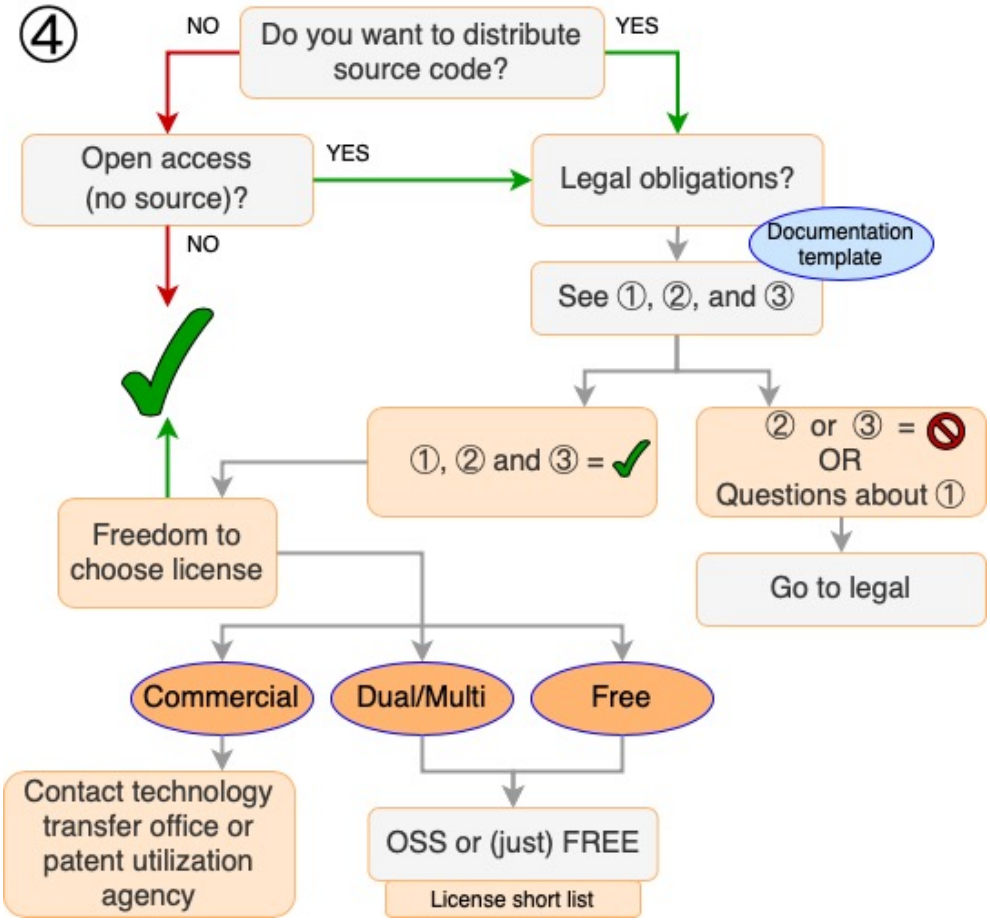

Figure 4. Licensing. Depending on the distribution model, open access (OA) or open source software can be selected. If you have already distributed the code, document distribution using Table 5

Table 5. Distributions of the code to the present day.

SW Name	License type	License name	Download location	Receiving party (closed/open)
⋮				

3 Acknowledgments

The paper is the result of a community effort, with work undertaken during two workshops and subsequent collaborative work across the larger RSE community in Germany. It has been initiated during a half-day workshop at the first International Conference for Research Software Engineers in Germany (deRSE19) in Potsdam, Germany on June 5th, 2019, and continued during a dedicated two-day workshop in Berlin, Germany on November 7th and 8th, 2019, which was funded by the DFG. We thank Bernhard Renard for his contribution in

organizing the workshops in Berlin and his contribution to this manuscript. We decided to use the alphabetical order of first names as sequence.

References

[1] H Anzt, F Bach, S Druskat, F Löffler, A Loewe, BY Renard, G Seemann, A Struck et al. "An environment for sustainable research software in Germany and beyond: current state, open challenges, and call for action", *F1000Research*, 2020, 9:295

[2] https://www.dfg.de/download/pdf/foerderung/rechtliche_rahmenbedingungen/gute_wissenschaftliche_praxis/kodex_gwp_en.pdf