Defining Research Software: a controversial discussion

Summary Report of FAIR4RS Subgroup 3 activity and discussion

Morane Gruenpeter (Inria, Software Heritage), Daniel S. Katz (University of Illinois), Anna-Lena Lamprecht (Utrecht University), Tom Honeyman (Australian Research Data Commons), Daniel Garijo (Information Sciences Institute), Alexander Struck (Cluster of Excellence Matters of Activity, de-RSE), Anna Niehues (Radboud university medical center), Paula Andrea Martinez (Research Software Alliance), Leyla Jael Castro (ZB MED Information Centre for Life Sciences), Tovo Rabemanantsoa (French National Research Institute for Agriculture, Food and Environment), Esther Plomp (Delft University of Technology - Faculty of Applied Sciences), Neil Chue Hong (Software Sustainability Institute / EPCC, University of Edinburgh), Carlos Martinez-Ortiz (Netherlands eScience Center), Laurents Sesink (Leiden University Libraries), Matthias Liffers (Australian Research Data Commons), Anne Claire Fouilloux (University of Oslo, Department of Geosciences), Chris Erdmann (American Geophysical Union), Silvio Peroni (University of Bologna), Paula Martinez Lavanchy (TU Delft Library), Ilian Todorov (UKRI STFC), and Manodeep Sinha (Swinburne University of Technology)

Version 1

Abstract

Software is essential in modern research; it plays vital roles at multiple stages of the research lifecycle. The term Research Software is widely used in the academic community but, what do we mean when we use these terms? Software and research? When you think of software, you may think of a digital object that is executed on a machine. Yet software is more than just this, it is a complex and evolving artifact. It may be a concept or a project designed to solve a puzzle by a team or a community that develops its functionalities and algorithms, which might not be digital objects. Furthermore, the software artifacts are digital objects, e.g., executables and source code files for different environments. These digital artifacts, which are used in a scholarly setting, might be important in the research process, but should all these be considered Research Software?

This report is the result of a discussion examining the scope of the community definition of the FAIR principles for Research Software as part of the work in the FAIR for Research Software working group (FAIR4RS). We aim to clarify the scope of the FAIR principles by identifying which software artifacts the FAIR principles should apply to. This discussion portrayed a complex landscape of software uses in research and existing definitions that can help to better understand the complexity of different types of software in academia. Finally we determine the scope of the FAIR4RS with a short and concise definition of Research Software as a separate metaphor of software in research.

Keywords: research software, definition, academic software, scientific software, software source code, FAIR principles

---

Table of contents

Introduction

The role of software in research is undeniably vital, while recommended best practices that are applied to other research objects (e.g., research data) are not always specifying the applicability to software artifacts. For this reason, the FAIR guiding principles (Wilkinson et al. 2016) originally specified for scientific data are being considered for software and other digital research objects, too. The FAIR for Research Software (RS) working group (FAIR4RS WG), launched in 2020, has taken on the task of producing a community definition of the FAIR principles for RS. During the first stage of the FAIR4RS WG, the Steering Committee coordinated four subgroups that could be worked on independently. This report documents the results from subgroup 3 working on the RS Definition, with the initial aim of deriving a concise community definition to specify the scope of the FAIR principles for RS.

Subgroup 3 started its work in July 2020 with the intention of providing a concise community definition of research software to provide scope for FAIR software principles. The collaborators have collected quotes from the literature (in Appendix A), collected software examples and discussed these examples in a collaborative document. In January and February 2021, subgroup 3 convened for two short calls and two workshops to gather feedback. The calls and workshops are documented and available in (FAIR4RS, 2021).

Agreeing on a concise definition was more challenging than expected, and therefore, we agreed on capturing the fascinating discussion and defining the scope in a more flexible manner. This report contains the following sections, reflecting our discussion:

- Why we need a definition
- Summary of collected quotes
- How to identify Research Software
- Raising awareness

Identifying the controversy in academia regarding what is or is not Research Software is in itself a big step forward in the process of developing a viable solution for specifying the scope of the FAIR principles for RS.

Why we need a definition

Scope for FAIR principles for Research Software

It is clear that setting the scope is a way of clarifying the usage intent of the proposed FAIR principles, which are specifically designed for RS. **Identifying the object of application** for the FAIR4RS principles is necessary in the process of agreeing on the community-driven principles for RS. We should be wary when defining RS and the scope for the FAIR4RS principles because the impact of this work can affect institutional policy and beyond.

“Research Software” is commonly used to refer to software used and/or generated in a research context, including and not limited to scientific, non-scientific, commercial, academic and non-academic research. Our definition should refer to objects to which the FAIR principles should apply to. Furthermore, software is an important component when it comes to reproducibility, where a different team needs to use the same research outputs and obtain the same results to validate and build upon this research.

What is Software?

When defining RS, one can start by defining software, since software is all around us and may take different forms: it can be available in different forms, as source code (also readable by humans), as executables (executable in a particular execution environment), or as software services, where the developed software is hosted on an online platform. Some of these forms might require hosting platforms that are accessible through a certain defined interface (e.g, application programming interfaces / APIs) or protocols. The Encyclopædia Britannica defines software as: *“Software, **instructions** that tell a computer what to do. Software comprises the entire **set of programs**, **procedures**, and **routines** associated with the operation of a computer system. The term was coined to differentiate these instructions from hardware—i.e., the physical components of a computer system”*[1]. However, this definition overlooks the “social” aspect of the software and focuses on its “technical” aspect.

Software can refer to the software project, which is complex as seen in (Alliez et al. 2019) it may vary in:

- **Structure**

[1] Access date: November 18, 2020
https://www.britannica.com/technology/software

    - monolithic / composite;
    - self-contained / external dependencies
- **Lifetime**
    - one-time / long term
- **Community**
    - one person / one team / distributed community
- **Authorship**
    - complex set of roles
- **Authority**
    - institutions / organizations / communities / single person

This complexity makes it challenging to agree on a single short definition, yet capturing the differences from a diverse set of disciplines merits an attempt to find consensus. This involves clarifying the complex relationship among the software creators, the software users, and the pieces of software themselves. It is difficult to take into consideration all software in all forms of delivery to which it might be important to apply FAIR principles, therefore we need to limit the scope of what Research Software is.

Summary of collected quotes

Semantic expressions used in the quest to define software and Research Software

To fully understand how different communities *perceive* Research Software, we need to examine the terminology and semantics used to define it. In Table 1, we identified semantically expressions used in the collected quotes, the full collection of quotes is available in Appendix A. Some semantic expressions are recurrent and focus on the research process, such as:

- Discovery
- Answer a question/problem
- Experimentation
- Act on research data (manage, analyze, produce, etc.)
- Research intent
- Present results

Table 1: Semantic expressions and their provenance

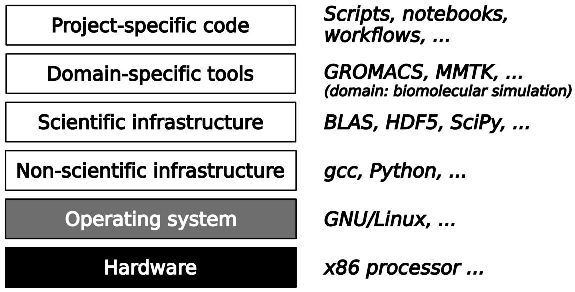
Software/ Research software semantics	Topic	Quoted in
set of instructions	General	(Chan, 2005) ESIP research artifact citation cluster
operating information	General	Oxford dictionary
what software does (rather than what software is)	General	(Matthews et al., 2010)
scientific discovery discovery process exploratory process facilitating a clear scientific workflow	Discovery	(Hasselbring et al., 2020) (Johanson & Hasselbring, 2018) (usgs.gov site, accessed 2020)
to answer a scientific question solves complex modeling problems model simulations	Answer a question/problem	(Kelly, 2011) (JOSS guidelines) (Gomez-Diaz & Recio, 2019) (Hasselbring et al., 2020)

scientific contribution Scientific integrity		(usgs.gov site, accessed 2020)
experimental apparatus make predictions about real world processes replace (or augment) physical experimentation execution of research experiments	Experimentation	(Wilson et al., 2015) (Kanewala & Bieman, 2014) (Heaton & Carver, 2015) (Kelly, 2011) (JOSS guidelines)
developing new algorithms implementing scientific algorithms	Research intent	(Wilson et al., 2015) (usgs.gov site, accessed 2020)
data analytics : - managing data - analyzing data - combining data producing scientific data provides data to be examined	Act on research data	(Wilson et al., 2015) (Hasselbring et al.2020) (Prlić et al., 2012) (usgs.gov site, accessed 2020)
collect observations	Act on research data	(Prlić et al.,2012)
extracts knowledge	Act on research data	(JOSS guidelines)
present results yielding scientific results	Present results	(Prlić et al.,2012 (usgs.gov site, accessed 2020))
product of your research	Research intent	(Wilson et al., 2017)
Intention (intended for research and researchers) intention to be used for research	Research intent	(Katz, 2015) (Soch, 2020)
used in science and engineering fields used for scientific purposes used to produce a result published or disseminated	Research intent	(Kanewala & Bieman, 2014) (Gomez-Diaz & Recio, 2019)
researchers develop to aid their science	Research intent	(National Academies of Sciences, Engineering, and Medicine, 2018)
used as evidence [in publications]	Present results	(Kanewala & Bieman, 2014)
require(s) specialized domain knowledge an expert in its scientific domain	Experimentation	(Kanewala & Bieman, 2014) (Kelly, 2011)
To better understand	Answer a question/problem	(Kanewala & Bieman, 2014)
people who write, maintain and manage this research software… critically important members of research teams written by a well identified research team	Research intent	(Cohen et al., 2020) (Gomez-Diaz & Recio, 2019)
Software that researchers ... may feel the need to have scholarly infrastructure support for	Research intent	(European Commission, 2020)
Reproducibility	Research intent	(usgs.gov site, accessed 2020)

Hinsen's stack: A cipher to identify Research Software?

Hinsen characterizes software created and used in research as a series of interdependent layers arranged in a 'stack' (Hinsen, 2019). The software stack was devised to show the importance of all parts of the stack for reproducibility, to prevent what Hinsen calls a "software collapse". In doing so he shows the many ways in which the (research) software at the topmost layers is fragile and dependent on the layers beneath. Moreover, some parts of the "Hardware" layer are in a transition phase to become more software-defined. Examples include virtual machines, containerization, software-defined networks and storage. Such technologies may facilitate application of the FAIR principles in some disciplines but also complicate the definition of (what belongs to) RS.

In the software stack there are six layers describing components that are needed to reproduce an experiment (Hinsen, 2019):



1. “...software written by scientists for a specific research project...scripts, notebooks, and workflows, but also special-purpose libraries and utilities.”
2. “...domain specific research software. These are tools and libraries that implement models and methods which are developed and used by communities ranging in size from a single research lab to thousands of researchers”
3. “...infrastructure created specifically for scientific computing, but not any particular domain.”
4. “Infrastructure software that is not specific to scientific computing. … compilers and interpreters, libraries for data management, but also higher level tools such as text editors and Web browsers. … obtain[ed] from the wider non-scientific software market”
5. operating system
6. hardware

The semantically strong expressions captured in Table 1 can be overlayed onto the Hinsen’s stack to visualize which expressions can be applied in one or more layers. This overlay is captured in Figure 2 below. When considering RS as software that is produced in the research lifecycle, the FAIR principles can be applied at the point of dissemination of the outputs. On the other hand, when looking at software that was used to perform research and is necessary for the reproducibility of the research, then the point of view shifts to the perspective of a user who wishes to have FAIR, citable software at hand.

To help understand the differences in points of view, we decided to use a spectrum between two types of definitions: an inclusive definition, which is closer to a usage point of view, and an exclusive definition, which is closer to a creation point of view:

- **Inclusive definition of Research Software**
    - All code and software artifacts that are used, produced, or might be related to the research process in one or more stages of the research lifecycle and regardless of the layer of the software stack.
    - Software that was not necessarily developed with the intention of being part of research, for example, a library for interfacing with a sensor, or software that ceased to be exclusive to the research domain, for example, certain programming languages developed in research projects, e.g., Python, Scala, R.
- **Exclusive definition of Research Software**
    - Well identified software that is part of the research discovery process, which might require specialized domain knowledge and is by itself a contribution to science and research.
    - Software that was developed with the intention of being part of research.

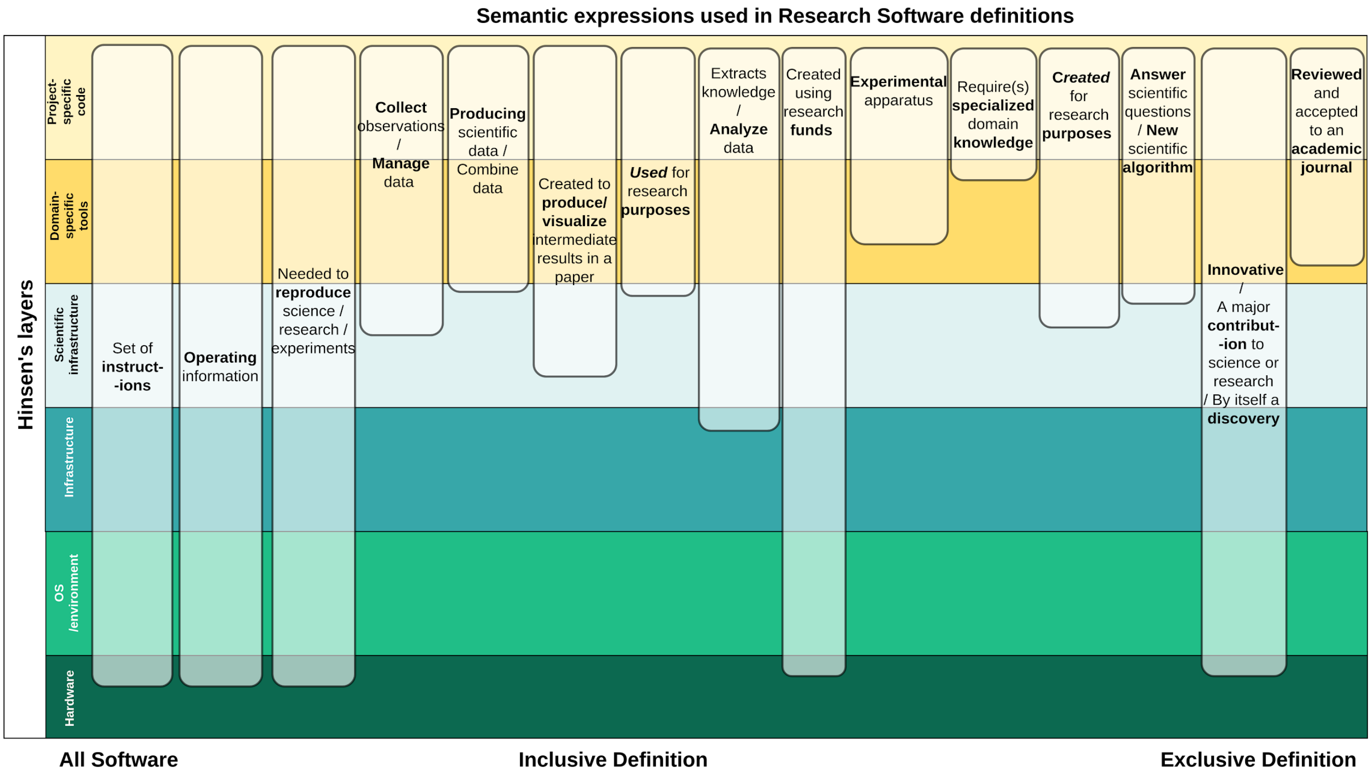


Figure 2: semantic expressions compared with Hinsen's layer visualized on a spectrum between all software, inclusive definition and exclusive definition

During the discussions, different views were presented around software that straddles the boundaries between scientific and non-scientific infrastructure, and whether "research" should include civilian, industrial and military research. We may consider the two definitions, inclusive and exclusive, as the far ends of a spectrum on which we want to identify a threshold as the limit where software is regarded as RS and should be FAIR. When deciding on this threshold we should be vigilant, because applying the FAIR principles for identified RS might become a requirement from academic institutions for FAIRification processes to be applied to. The readers from different disciplines may use the definition (inclusive or exclusive) that fits best their community standards and practices, while keeping in mind that software is a complex and living object, which is difficult to develop and maintain and any extra requirements should be chosen cautiously.

How to identify Research Software?

Analysis of questions

What can be considered as RS is difficult to agree upon, since usage of software is so abundant everywhere, including in research contexts. Should software used to write an article, e.g., Microsoft Word or LaTeX, be considered RS? The same can be asked of software used to capture data, which might also be used to analyze and process data, like Microsoft Excel: should this be identified as Research Software? Or do we exclude all usage of software and only focus on the software that was produced during research? Should it be limited to implementing innovative algorithms, or new implementations of existing algorithms? Should it include software written for research processes? Should these processes be an exclusive set, or an open set including anything related to research work? In the process of identifying RS, we have discussed potential questions that can be helpful to determine if the software at hand is RS. In Table 2 we summarize the questions and some of the sentences captured during the discussions; the text in the bullet points correspond to text added directly by participants to the collaborative notes during the discussion sessions.

Table 2: List of questions and discussion to see how helpful this Q&A is when determining that software is RS

Question	Summary of discussion during the workshops
Is the software created with a research or commercial purpose?	● Not helpful. False dichotomy… can be both research and commercial. ● Not helpful, as research happens everywhere (industry, military, academy). ● Not helpful. It raises more questions than providing clarity. ● I would not oppose research and commercial or maybe we should be more precise about what we mean by “research”. ● Hardly helpful when the software was a research result a long time ago and is definitely used in industry today. ● It helps to understand the context a bit.
Is the software primarily used for purposes outside of research?	● Partly helpful as it is difficult to tell from the paper where it was or should be used. ● Not helpful. It may start as research-bound, and then become applicable outside research. ● Over the years the use cases may have changed from purely academic to industry now.
Is the software/code needed for reproducing data analysis/results? deliver research results?	● Seems very narrow. Although I might have this requirement (reproduction of results) outside a research context too. ● Not necessarily, Python is used to reproduce results, I would not categorize it as RS (nowadays).
Is there a paper about the software?	● Yes. It is helpful to determine whether this is RS, but I don’t think it should be a requirement. ● Yes, because it means that the software was the scope of research. ● This seems overly restrictive. ● This is a bit “old” fashion and linked to how “researchers” are evaluated. However, there is always a need to communicate about the software (paper or any other communication). ● Yes, if it’s a software-related academic text publication and this would be helpful for a definition work. ● Yes, a paper is a good way to evaluate as RS. ● Seems too narrow. ● Yes, but software without a paper could also be RS.

	• Helpful if we agree that a peer-reviewed paper is a valid evaluation proxy for a piece of software. • Yes, I would say that gives the idea that, at least at some point in time, research was the aim… so, maybe programming languages can also be RS.
Is there a research result that is dependent on this software?	• Any research result? If the software is a research result? • Hardly helpful as we do not yet know what to in/exclude from the software stack. • Lot of research results depend on Python, would Python (the language) be considered RS? • I'm just not sure what is meant by "dependent" here. • That is hard to tell because with the case of a programming language inherently created to build tools, the same algorithm in a different language may solve the same problem. So, in the case of programming languages, this question is not helpful. • Difficult to say.
Could the software be replaced with other software without affecting the research results?	• If I have access to the source code, or a detailed explanation of an algorithm, then in theory, any software could be re-implemented. • Hardly helpful because cluster algorithms are supposed to be general purpose (network research) tools. • Do not see how it helps. • Not Helpful. This seems like a more useful criterion for determining the uniqueness of a tool.
Was the software developed by a researcher? Or in an academic context?	• Somehow useful, but it shouldn't be the only criterion taken into account. • Helpful, it makes clear that the software was developed in an academic context. • Yes, this looks like a typical example. • Helpful, except to clarify "researcher", and "academic context" seems overly restrictive. • Not helpful. What about research institutes that are not necessarily part of an academic institute and industry research teams? • Not helpful as some algorithms used in network research have been developed by the military or by military funding (Example Ward Clustering). Should we consider that "academic"? • Depends on the definition of "researcher".
Was the software evaluated or reviewed as part of a research process? (ACM badge[2],	• Doesn't seem useful, unless you want to define RS exclusively as a scholarly output. • Not much, not sure how software reviewing is required/done for research… It is important, yes, but detailed review can take too long

[2] ACM- Association of Computing and Machinery
https://www.acm.org/publications/policies/artifact-review-and-badging-current

peer review, research institution committee, etc.)	and reviewers are commonly not paid for it so not sure how much time would be dedicated to go to all the details. I would say metadata (e.g., the one needed for FAIR) would be easier to "review". ● Not sure. It seems to be widely used, so that is an evaluation by the wide public maybe? ● Reviewed by academic peers can be a useful question.
Would a researcher be expected to cite this software?	● Not sure if useful, software citation hasn't been picked up widely. ● Helpful. Citation could be a result of software being FAIR. ● Yes. ● Yes. Due to common practice to document what your research is based on, a citation (expectation/recommendation) may be used as an indicator for something being a research result. ● Yes, (at that time) it would give clues of this to be an RS. ● Yes, credit is the way authors claim their product is research.

A list of helpful elements to search within the software that might provide insight on its academic intentions:

- Citation information (in README or a codemeta.json file or a CITATION.cff)
- A related publication describing the software
- A badge (e.g., Association for Computing Machinery- ACM badges [3])
- The intent of being created and disseminated on scholarly infrastructures

Analysis of real life software

In this section, we analyze three cases of software in research to determine whether they correspond to RS. We describe their nature (source code, executable, project, etc.) and determine which granularity level of identification is used with each use case. Also, we document the creators of the software and answer the questions identified in the previous section. We introduce a table for each analyzed case of software, see Tables 3, 4 and 5. To avoid repetition, we have selected some representative comments for those points where we observed a high degree of agreement. We also captured disagreement by including (possibly) conflicting points of view. All the comments were gathered during the RS workshops in February 2021.

IPOL Journal publication

The following example is a published article in the Image Processing On Line (IPOL) journal. On the publication webpage there is the possibility to view the article and a demo. There is also a link to access the source code on Software Heritage with a SoftWare Heritage persistent IDentifier (SWHID) and the possibility to directly download the source code. The downloadable source code is a directory and the SWHID is also identifying the directory granularity level.

In this example the creators of the software are identified and cited on the webpage and in the article.

[3] https://www.acm.org/publications/policies/artifact-review-and-badging-current

Figure 1: View of a publication on the IPOL Journal platform (Lisani, 2018)

Table 3: List of questions and answers for IPOL

Questions	**Answers** for **IPOL published implementation (from the RS workshops)**
Is the software created with a research or commercial purpose?	• Research, which does not mean that it cannot be used as commercial.
Is the software primarily used for purposes outside of research?	• The paper lacks research motivation. We see some grants but the purpose sounds like improvement of a tool • Not enough information.
Is the software/code needed for reproducing data analysis/results? deliver research results?	• Yes, if it is for images, which are data. • The software is needed for reproducing the analysis of itself
Is there a paper about the software?	• Yes.

Is there a research result that is dependent on this software?	• Unknown • We can imagine so, at least a contrast between the two implementations.
Could the software be replaced with other software without affecting the research results?	• Needs more investigation. • Yes, this looks like it because it is an implementation. • No, because the results of the research are about the software itself.
Was the software developed by a researcher? Or in an academic context?	• Yes, corresponding author is at a university • Yes. • Not sure. We see grants but do not know if they are military or academic. • Yes, you can look it up in the article.
Was the software evaluated or reviewed as part of a research process? (ACM badge, peer review, research institution committee, etc.)	• Unknown. One would need to look up IPOL's peer review process. • No idea, reviewing the paper is not the same as reviewing the software.
Would a researcher be expected to cite this software?	• Yes, recommendation is to cite the associated paper. • Yes, because common in research, • Yes, it looks like it. • ... they could cite the paper or the software… so strictly speaking no.

SciPy

Here an example of a software entry in the swMath registry for mathematical software. On the swMath webpage (https://swmath.org/) there is the possibility to view the software metadata and the list of articles where the software was cited. There is also a link to access the source code on Software Heritage. The granularity level of this entry is at the project level, since the version is not specified.
In this example the creators of the software are identified and cited in the metadata.

Figure 2: SciPY entry in swMATH registry which shows an metadata about SciPy and the list of citations where SciPy was cited (492 articles) accessed on February 22, 2021

Table 4: List of questions and answers for SciPy

Questions	**Answers on swMath entry- SciPy (from the RS workshops)**
Is the software created with a research or commercial purpose?	• Created with research purpose but used for both research and commercial purposes
Is the software primarily used for purposes outside of research?	• I would say yes with 80% certainty
Is the software/code needed for reproducing data analysis/results? deliver research results?	• Yes
Is there a paper about the software?	• Yes, multiple papers

Is there a research result that is dependent on this software?	● Yes
Could the software be replaced with other software without affecting the research results?	● Yes, most of the functionality of this software can be translated to other programming languages
Was the software developed by a researcher? Or in an academic context?	● Scipy is an ecosystem thus it was developed by a community, and in research context
Was the software evaluated or reviewed as part of a research process? (ACM badge, peer review, research institution committee, etc.)	● Review as the usage of scipy by millions of users (better than an "old" fashion review).
Would a researcher be expected to cite this software?	● Yes, that's why citation is provided

Scala

Scala is an example of a very popular programming language which started as a research project in a research institution. The granularity level of this entry is at concept level, since this project has evolved enormously in its lifespan. In this example the creators of the software are identified in the article, but there may have been many contributions after this initial paper.

An Overview of the Scala Programming Language

Second Edition

Martin Odersky, Philippe Altherr, Vincent Cremet, Iulian Dragos
Gilles Dubochet, Burak Emir, Sean McDirmid, Stéphane Micheloud,
Nikolay Mihaylov, Michel Schinz, Erik Stenman, Lex Spoon, Matthias Zenger

École Polytechnique Fédérale de Lausanne (EPFL)
1015 Lausanne, Switzerland

Technical Report LAMP-REPORT-2006-001

Abstract

Scala fuses object-oriented and functional programming in a statically typed programming language. It is aimed at the construction of components and component systems. This paper gives an overview of the Scala language for readers who are familar with programming methods and programming language design.

1 Introduction

True component systems have been an elusive goal of the software industry. Ideally, software should be assembled

guage for component software needs to be scalable in the sense that the same concepts can describe small as well as large parts. Therefore, we concentrate on mechanisms for abstraction, composition, and decomposition rather than adding a large set of primitives which might be useful for components at some level of scale, but not at other levels. Second, we postulate that scalable support for components can be provided by a programming language which unifies and generalizes object-oriented and functional programming. For statically typed languages, of which Scala is an instance, these two paradigms were up to now largely separate.

To validate our hypotheses, Scala needs to be applied

Figure 4: A combined figure from the first page of the first Scala publication (cited 782 times accessed on google scholar on February 22, 2021) and a diagram explaining scala (Odersky, 2004).

Table 5: List of questions and answers for Scala

Questions	Answers for the Scala programming language (from the RS workshops)
Is the software created with a research or commercial purpose?	● Others can now create RS with it. ● At that point in time (2008) it was RS and created with research intention.
Is the software primarily used for purposes outside of research?	● Not aware.
Is the software/code needed for reproducing data analysis/results? deliver research results?	● Yes.
Is there a paper about the software?	● Yes.
Is there a research result that is dependent on this software?	● Maybe not, assuming you mean the result question that began the work, not things written in Scala. ● Yes, any RS written in Scala.
Could the software be replaced with other software without affecting the research results?	● Hard to say, I guess yes, I guess other programming languages can have similar features.
Was the software developed by a researcher? Or in an academic context?	● Yes, originally, and then… ● Yes, looks like that at that point. ● Yes.
Was the software evaluated or reviewed as part of a research process? (ACM badge, peer review, research institution committee, etc.)	● No, but it looks like the paper was reviewed. ● Yes, the programming language was evaluated.
Would a researcher be expected to cite this software?	● No. ● Yes, at that time at least.

To answer the question, is Scala RS? The answer can be it started as RS and became a popular and important programming language used worldwide. The current versions of Scala aren't developed as part of research experiments and shouldn't be considered RS. When software is created as RS it will always be born as RS. The evolution of it being RS might depend on version x.z and if this version is also born RS.

Raising Awareness

The complexity of software and the multiple facets it can play in research is evident. RS is an eluding concept where some artifacts can be identified as RS in specific disciplines and not in others.

There is a strong intertwined relationship between the software creators, the software users, and the software as an artifact, which may lead us into believing that all software created or used in academia is RS. However, for the FAIR principles for RS, following an “inclusive” interpretation might create unnecessary requirements for all software developed or used in research, while following an “exclusive” interpretation might leave important pieces of software out of academia's consideration. The following definition is an attempt to consolidate the different approaches and provide a flexible solution:

Research Software includes source code files, algorithms, scripts, computational workflows and executables that were created during the research process or for a research purpose. Software components (e.g., operating systems, libraries, dependencies, packages, scripts, etc.) that are used for research but were not created during or with a clear research intent should be considered software in research and not Research Software. This differentiation may vary between disciplines. The minimal requirement for achieving computational reproducibility is that all the computational components (Research Software, software used in research, documentation and hardware) used during the research are identified, described, and made accessible to the extent that is possible.

Acknowledgements

We would like to acknowledge the participants of subgroup3 of the FAIR4RS working group (a complete list is available in Appendix B) for the lively discussion we had during several group calls, including two dedicated workshops.
We would like to acknowledge the participants of the FAIR4RS who reviewed the document and helped improve the text (a complete list is available in Appendix C).

Morane Gruenpeter acknowledges the support of the FAIRsFAIR project (European Commission grant 831558).
Alexander Struck acknowledges the support of the Cluster of Excellence »Matters of Activity. Image Space Material« funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – EXC 2025 – 390648296.
Anna Niehues acknowledges support from EATRIS-Plus, which has received funding from the European Union's Horizon 2020 Research and Innovation programme under grant agreement No 871096, and from the Netherlands X-omics Initiative, which is (partially) funded by the Dutch Research Council (NWO), project 184.034.019.

References

- Pierre Alliez, Roberto Di Cosmo, Benjamin Guedj, Alain Girault, Mohand-Said Hacid, et al.. Attributing and Referencing (Research) Software: Best Practices and Outlook from Inria. Computing in Science and Engineering, Institute of Electrical and Electronics Engineers, 2019, pp.1-14. ⟨10.1109/MCSE.2019.2949413⟩. ⟨hal-02135891v2⟩
- Bechhofer, S., De Roure, D., Gamble, M. et al. Research Objects: Towards Exchange and Reuse of Digital Knowledge. Nat Prec (2010). https://doi.org/10.1038/npre.2010.4626.1
- Chun: On Software, or the Persistence of Visual Knowledge, 2005. https://doi.org/10.1162/1526381043320741
- Cohen et al. "The Four Pillars of Research Software Engineering," in IEEE Software, doi: https://doi.org/10.1109/MS.2020.2973362

- FAIR4RS, subgroup3. (2021). Research Software Definition workshop. Zenodo. https://doi.org/10.5281/zenodo.5503982
- Gomez-Diaz T and Recio T. On the evaluation of research software: the CDUR procedure [version 2; peer review: 2 approved]. F1000Research 2019, **8**:1353 (https://doi.org/10.12688/f1000research.19994.2)
- Hasselbring et al.: From FAIR research data toward FAIR and open research software, 2020. https://doi.org/10.1515/itit-2019-0040
- D. Heaton, J. Carver: Claims about the use of software engineering practices in science: A systematic literature review, 2015, https://doi.org/10.1016/j.infsof.2015.07.011
- K. Hinsen, "Dealing With Software Collapse," in Computing in Science & Engineering, vol. 21, no. 3, pp. 104-108, 1 May-June 2019, doi: 10.1109/MCSE.2019.2900945.
- Software Deposit: Guidance for Researchers Michael Jackson (ed.), The Software Sustainability Institute Version 1.0 doi:10.5281/zenodo.1327310
- Michael Jackson (ed.) (10 December 2018). Checklist for a Software Management Plan (Version 1.0). Zenodo. doi:10.5281/zenodo.2159713. Web site: https://www.software.ac.uk/softwaremanagement-plans
- A. Johanson, W. Hasselbring: Software Engineering for Computational Science: Past, Present, Future, 2018, https://doi.org/10.1109/MCSE.2018.021651343
- U. Kanewala, J. Bieman: Testing scientific software: A systematic literature review, 2014, https://doi.org/10.1016/j.infsof.2014.05.006
- Kelly D: An Analysis of Process Characteristics for Developing Scientific Software. J Organ End User Com. 2011; **23**(4): 64–79. https://doi.org/10.4018/joeuc.2011100105
- Scientific Software Challenges and Community Responses (Daniel S. Katz)
- https://www.slideshare.net/danielskatz/scientific-software-challenges-and-community-responses
- Jose-Luis Lisani, An Analysis and Implementation of the Shape Preserving Local Histogram Modification Algorithm, Image Processing On Line, 8 (2018), pp. 408–434. https://doi.org/10.5201/ipol.2018.236
- Matthews et al: A Framework for Software Preservation, 2010. https://doi.org/10.2218/ijdc.v5i1.145
- National Academies of Sciences, Engineering, and Medicine: Open Source Software Policy Options for NASA Earth and Space Sciences. Washington, DC: The National Academies Press. 2018. https://doi.org/10.17226/25217
- Odersky, M., Altherr, P., Cremet, V., Emir, B., Maneth, S., Micheloud, S., ... & Zenger, M. (2004). An overview of the Scala programming language. Technical Report IC/2004/64
- Prlić et al. Ten simple rules for the open development of scientific software. PLoS Comput Biol. 2012;8(12):e1002802. Doi: https://doi.org/10.1371/journal.pcbi.1002802
- G. Wilson et al.: Best practices in scientific computing, 2015, https://doi.org/10.1371/journal.pbio.1001745
- G. Wilson et al.: Good enough practices in scientific computing, 2017, https://doi.org/10.1371/journal.pcbi.1005510

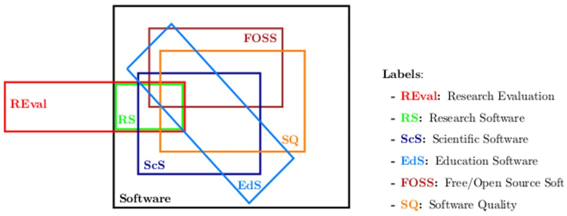
Appendices

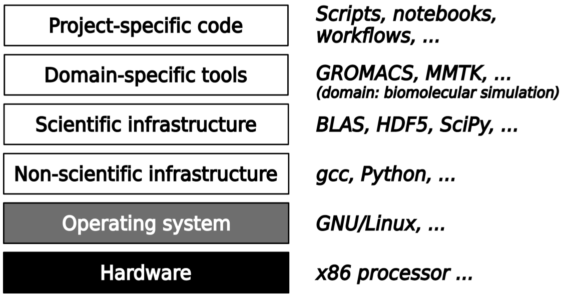
Appendix A: Collection of quotes

Quote(s)	tags	Source
“a **set of instructions** that direct a computer to do a specific task”	[Software]	Chun: On Software, or the Persistence of Visual Knowledge, 2005. https://doi.org/10.1162/1526381043320741
“A **set of instructions** that performs some action, either as **source code** (machine-readable) or **executable**”	[Software]	https://wiki.esipfed.org/Research_Object_Citation
“Research software is software that is **employed** in the **scientific discovery process** or a **research object** itself. Computational science (also scientific computing) involves the development of research software for **model simulations** and **data analytics** to understand natural systems answering questions that neither theory nor experiment alone are equipped to answer. Computational science is a multidisciplinary field lying at the intersection of mathematics and statistics, computer science, and core disciplines of science and research.”	[Research Software]	Hasselbring et al.: From FAIR research data toward FAIR and open research software, 2020. https://doi.org/10.1515/itit-2019-0040
“we are more interested in **what software does** rather than **what software is**.”	[Software]	Matthews et al: A Framework for Software Preservation, 2010. https://doi.org/10.2218/ijdc.v5i1.145
“Software is as important to modern scientific research as telescopes and test tubes. [...], more and more of the daily operation of science revolves around **developing new algorithms**, **managing** and **analyzing** the large amounts of **data** that are generated in single research projects, **combining** disparate **datasets** to assess synthetic problems, and other **computational tasks**. Scientists typically develop their own software for these purposes because doing so requires substantial **domain-specific knowledge**.” “We believe that software is just another kind of **experimental apparatus**” “Unlike traditional commercial software developers, but very much like developers in open source projects or startups, scientific programmers usually don't get their requirements from customers, and **their requirements are rarely frozen**.”	[Research Software]	G. Wilson et al.: Best practices in scientific computing, 2015, https://doi.org/10.1371/journal.pbio.1001745

"If you or your group are **creating tens of thousands of lines of software** for use by hundreds of people you have never met, you are doing **software engineering**. If you're writing a few dozen lines now and again and are probably going to be their only user, you may not be doing engineering, but you can still make things easier on yourself by adopting a few key **engineering practices**." "Your software is as much a **product of your research** as your papers [...]"	[Research Software]	G. Wilson et al.: Good enough practices in scientific computing, 2017, https://doi.org/10.1371/journal.pcbi.1005510
"We have a pretty strong sense of what constitutes software - it is some kind of **compiled or interpreted program that is run by a computer**. "	[Software]	https://vsoch.github.io/2020/what-is-research-software/
Software "The programs and other operating information **used by a computer**."	[Software]	Oxford dictionary
"Scientific software is widely **used in science and engineering fields**. [...] In addition, results from scientific software are **used as evidence** in research publications. Due to the **complexity** of scientific software and the required **specialized domain knowledge**, scientists often develop these programs themselves or are closely involved with the development." "We define scientific software broadly as software **used for scientific purposes**. Scientific software is mainly developed to **better understand** or **make predictions about real world processes**. The size of this software ranges from 1,000 to 100,000 lines of code. Developers of scientific software range from scientists who do not possess any software engineering knowledge to experienced professional software developers with considerable software engineering knowledge."	[Research Software] [Software in Research]	U. Kanewala, J. Bieman: Testing scientific software: A systematic literature review, 2014, https://doi.org/10.1016/j.infsof.2014.05.006
"Scientists and engineers often use computational modeling to **replace (or augment) physical experimentation**. For the remainder of this paper we will refer to the software created by these scientists and engineers as scientific software."	[Research Software]	D. Heaton, J. Carver: Claims about the use of software engineering practices in science: A systematic literature review, 2015, https://doi.org/10.1016/j.infsof.2015.07.011
"the variety of scientific software and its applications is large" "scientific software is an integral part of a **discovery process**"	[Scientific Software]	A. Johanson, W. Hasselbring: Software Engineering for Computational Science: Past, Present, Future, 2018,

"scientific software is deeply embedded into an **exploratory process**"		https://doi.org/10.1109/MCSE.2018.021651343
"While software has been a part of research for many decades, the **people who write, maintain and manage** this research software are increasingly seen as critically important **members of research teams**, rather than just "the people who write code"."	[Research Software]	Cohen et al. "The Four Pillars of Research Software Engineering," in IEEE Software, doi: https://doi.org/10.1109/MS.2020.2973362
"**Open-source software** development has had significant impact, not only on society, but also on **scientific research**. Papers describing software published as open source are amongst the most widely cited publications (e.g., BLAST [1], [2] and Clustal-W [3]), suggesting **many scientific studies may not have been possible without** some kind of open software to **collect observations**, **analyze data**, or **present results**. It is surprising, therefore, that so few papers are accompanied by open software, given the benefits that this may bring."	[Scientific Software]	Prlić et al. Ten simple rules for the open development of scientific software. PLoS Comput Biol. 2012;8(12):e1002802. Doi: https://doi.org/10.1371/journal.pcbi.1002802
Research software vs. infrastructure software • Some software is **intended for research** – Funded by many parts of NSF, sometimes explicitly, often implicitly – Intended for immediate use by developer – Maybe archived for future use and reproducibility • Other software is **intended as infrastructure** – Funded by many parts of NSF, often ACI, almost always explicitly – Intended for use by community • Focus mostly on infrastructure software, but many issues cross between – Reproducibility causes the most overlap	[Research Software] [Software in Research]	Scientific Software Challenges and Community Responses (Daniel S. Katz) https://www.slideshare.net/danielskatz/scientific-software-challenges-and-community-responses
*"Research software – that is, the software that **researchers develop to aid their science**..."*	[Research Software]	National Academies of Sciences, Engineering, and Medicine: Open Source Software Policy Options for NASA Earth and Space Sciences. Washington, DC: The National Academies Press. 2018. Publisher Full Text
"Scientific software is defined by three characteristics: *(1) it is developed **to answer a scientific question**;* *(2) it relies on the close involvement of **an expert in its scientific domain**; and* *(3) it **provides data to be examined** by the person who will answer that question ..."*	[Scientific Software]	Kelly D: An Analysis of Process Characteristics for Developing Scientific Software. *J Organ End User Com.* 2011; **23**(4): 64–79. Publisher Full Text

https://f1000researchdata.s3.amazonaws.com/manuscripts/23236/fe3e7601-5d98-4895-bdab-089a0580ab00_figure1.gif  *Not a quote, but a useful diagram representing a proposed common space between several overlapping concepts occupied by "research software"*	[Research Software]	Gomez-Diaz T and Recio T. On the evaluation of research software: the CDUR procedure [version 2; peer review: 2 approved]. *F1000Research* 2019, **8**:1353 (https://doi.org/10.12688/f1000research.19994.2)
"Research software [RS] is a well identified set of code that has been written by a well identified [...] **research team**. It is software that has been built and used to **produce a result published** or **disseminated** in some article or **scientific contribution**. Each [RS] encloses a set (of files) that contains the source code and the compiled code. It can also include other elements [such] as the documentation, specifications, use cases […]"	[Research Software]	Gomez-Diaz T and Recio T. On the evaluation of research software: the CDUR procedure [version 2; peer review: 2 approved]. *F1000Research* 2019, **8**:1353 (https://doi.org/10.12688/f1000research.19994.2)
*Software that **researchers** in any discipline may feel the need to have **scholarly infrastructure support for***, no matter if it is considered a **tool**, a **result** or an **object of study**	[Research Software] [Software in Research]	European Commission. Directorate General for Research and Innovation. (2020). Scholarly infrastructures for research software: report from the EOSC Executive Board Working Group (WG) Architecture Task Force (TF) SIRS. Publications Office. https://doi.org/10.2777/28598
Scientific software is made up of different but **interdependent layers** best considered part of an overall **stack**: (1) *"...software written **by scientists** for a specific **research project**...**scripts**, **notebooks**, and **workflows**, but also special-purpose **libraries and utilities**."* (2) "*...**domain specific** research software. These are tools and libraries that implement **models and methods** which are developed and used by communities ranging in size from a single research lab to thousands of researchers*" (3) "*...infrastructure created specifically for **scientific computing**, but not any particular domain.*"	[Research Software] [Scientific Software] [Software in Research]	K. Hinsen, "Dealing With Software Collapse," in *Computing in Science & Engineering*, vol. 21, no. 3, pp. 104-108, 1 May-June 2019, doi: 10.1109/MCSE.2019.2900945. **Figure**: Konrad Hinsen's Software Stack: A typical software stack in scientific computing consists of four layers on top of hardware and systems software. DOI: 10.7717/peerjcs.158/fig-1

(4) "*Infrastructure software that is not specific to scientific computing. … compilers and interpreters, libraries for data management, but also higher level tools such as text editors and Web browsers. … obtain[ed] from the wider non-scientific software market*" (5) operating system (6) Hardware 		
What is research software? The Software Sustainability Institute takes the view that research software is any software **used in research** and does not differentiate between what are often termed **scripts**, written in scripting languages such as bash shell or Python or R, and **programs**, written in "traditional" programming languages such as C, C++, Fortran or Java. In the view of the Institute a 50 line bash shell script for manipulating and filtering files, a collection of 50 line R scripts for running a bioinformatics analysis, 10,000 lines of Java for medical image analysis or 100,000 lines of Fortran using MPI for computational fluid dynamics are **all examples of research software and may be suitable candidates for deposit** into a digital repository. It is this view of research software that is assumed throughout the guides.	[Software in Research]	Software Deposit: Guidance for Researchers Michael Jackson (ed.), The Software Sustainability Institute Version 1.0 doi:10.5281/zenodo.1327310
"**Research Objects** [are] semantically rich aggregations of resources that bring together the **data**, **methods** and **people involved** in (scientific) investigation. [.....] ROs are both themselves resources accessible via linked data principles, and will aggregate linked data resources. [....] As an identifiable container, Research Objects allow us **to compute and attribute measure of trust to the object itself** [...]"	[Research Object]	Bechhofer, S., De Roure, D., Gamble, M. et al. Research Objects: Towards Exchange and Reuse of Digital Knowledge. Nat Prec (2010). https://doi.org/10.1038/npre.2010.4626.1
"The USGS Scientific Software Instructional Memo defines Scientific Software as: "Software containing source code **implementing scientific algorithms and/or producing scientific data**". It may include any custom developed code **yielding scientific results**, thereby, **facilitating a clear scientific**	[Scientific Software]	https://www.usgs.gov/products/software/software-management

workflow of analysis, **scientific integrity**, and **reproducibility**."		
JOSS publishes articles about research software. This definition includes software that: **solves complex modeling problems** in a scientific context (physics, mathematics, biology, medicine, social science, neuroscience, engineering); supports the **functioning of research instruments** or the **execution of research experiments**; **extracts knowledge** from large data sets; offers a **mathematical library, or similar**.	[Research software]	https://joss.readthedocs.io/en/latest/submitting.html#what-we-mean-by-research-software
● Research software is a subset of software ● Research software must be used by at least one researcher ● It might be developed by a research software engineer, but doesn't have to be ● It doesn't necessarily have to be intended for a particular domain ● Absence of **citation** does not disqualify it, but **presence strengthens it** ● Taking it away would be **a detriment to research** ● It was created with **intention to be used for research**	[Software in Research]	https://vsoch.github.io/2020/what-is-research-software/
Criteria and Taxonomy From our criteria discussion in the first section, we can derive the following questions: ● Is it software (all research software must be software) (yes/no) ● Is it used by at least one researcher? (yes/no) ● Is it developed by a research software engineer? (yes/no) ● Has it been cited in a research context (yes/no) ● Is it intended for **a particular scientific domain**? (yes/no) ● Would taking it away be **a detriment to research**? (yes/no) ● Was it created with the **intention** to be used for research? (yes/no)	[Software in Research]	https://vsoch.github.io/2020/what-is-research-software/

Research software can take **many guises**. It can be a 50 line bash shell script for manipulating and filtering files, a collection of 100 line R scripts for running a bioinformatics analysis, 10,000 lines of Java for medical image analysis or 100,000 lines of Fortran for computational fluid dynamics. It may be written in scripting languages such as Unix shell, Python, R or MATLAB or in "traditional" programming languages such as C, C++, Fortran or Java. But, whatever guise it takes, research software is **an integral part of the modern research ecosystem**.	[Research software] [Software in Research]	Michael Jackson (ed.) (10 December 2018). Checklist for a Software Management Plan (Version 1.0). Zenodo. doi:10.5281/zenodo.2159713. Web site: https://www.software.ac.uk/softwaremanagement-plans

Appendix B: FAIR4RS subgroup3 participants

Full name	**Affiliation**
Morane Gruenpeter (MG)	Software Heritage, Inria
Paula Andrea Martinez	Research Software Alliance
Daniel S. Katz (DSK)	University of Illinois
Neil Chue Hong (NCH)	Software Sustainability Institute / EPCC, University of Edinburgh
Anna-Lena Lamprecht (AL)	Utrecht University
Shoaib Sufi (S.Sufi)	Software Sustainability Institute, University of Manchester
Maggie Hellström	Lund University, Sweden and ICOS Carbon Portal
Silvio Peroni (SP)	University of Bologna / OpenCitations
Catherine Jones	Science and Technology Facilities Council (STFC)
Carlos Martinez-Ortiz	Netherlands eScience Center
Maria Cruz	NWO
Manodeep Sinha	Swinburne University of Technology
Alexander Struck	Cluster of Excellence Matters of Activity, de-RSE
Carlos Luis Parra-Calderón	FAIR4Health Project Coordinator
Ilian Todorov	UKRI Science and Technology Facilities Council, UK
Tom Honeyman (TH)	Australian Research Data Commons
Sergio Martinez Cuesta	University of Cambridge and AstraZeneca
Nicholas May	RMIT University
Daniel Garijo (DG)	Information Sciences Institute, USC
Anna Niehues (AN)	Radboud university medical center
Tovo Rabemanantsoa (TR)	French National Research Institute for Agriculture, Food and Environment - INRAE

Appendix C: FAIR4RS participants who reviewed and commented the document

Full name	**Affiliation**
Udayanto Dwi Atmojo	Aalto University
Limor Peer	Yale University
Chris erdman	American Geophysical Union
Nick Jones	New Zealand eScience Infrastructure
Nadica Miljković	University of Belgrade - School of Electrical Engineering
Bernadette Fritzsch	AWI
Jean-Luc Labordl	
Franco Donati	Leiden University CML
Guillaume Flandin	University College London