The role of Infrastructure for Software in Open Science

Ulrike Lucke
Let us first dive a bit deeper into the issues related to research software before we find out how a proper infrastructure might help us here. Software is a complex thing, and it can be different. It can be an artefact we create in our research, a tool we use in our research, or the subject of research itself.

The last perspective of those three is computer science, but let us have a closer look at the other two, which are relevant for others disciplines, too. Unfortunately, there is no common pattern that research software follows. Research software is as diverse as the disciplines are. Based on an overview of software used for research as analysed by the French Ministry of Research, there are only a few, rather generic tools broadly used (dominated by proprietary office software) and then a broad spectrum of other different software. Some are rather tools, meaning software that is used for research as it is. For others, researchers are also involved in the production of that software as a part of their research process. Both have their own challenges for sustainable maintenance of that software.

These challenges keep us from being effective at the moment. Long-term availability of research software is crucial for the reproducibility of research results, a key factor for quality assurance in research. However, many studies have revealed that reproducibility is often hindered by missing availability of the used software.

The results of a study carried out by Daniel Nüst and others in 2018 in the field of geosciences showed that even if data is available, there is little or no reproducibility of the overall results. Throughout the pipeline of research from input data, pre-processing and analysis of that data, over code to the computational environment and the final results, there is a leakage with the code or, latest, with its execution. Christian Collberg and his colleagues had already identified reasons for this leakage in 2014 for the field of computer systems. More or less the same picture exists today. If available at all, and this is still the main problem – software is not available, neither in an archive nor by email – the code of that software simply fails in terms of being built to an executable program or this programme is not executable anymore on a new platform.
Therefore, a finding could be that more emphasis should be put on the quality of developing, publishing and maintaining that software. The methodological basis of software engineering could help here. Software engineering is not the same as mere coding. It is more like a structured design process, and it could help to overcome such reproducibility issues. Software engineering is a professional activity requiring significant effort, training, and skills. It is not enough to just write 100,000 lines of code to deserve recognition, but it is the usability of that software by others that makes this a result to be recognised.

One could argue that we do not have such problems if we use existing tools; not to develop our own software, but to use existing ones and just rely on that basis. For instance, the presentation that I gave on this topic at the conference was designed with MS Office 2013 and I rely on that, aware that this might result in conflicts with other tools. Many developments of virtual reality applications in my group are developed in Unity, version 2019 at the moment, and they will hardly run with any other tool. Another example: one of the very top-ranked tools in the French Ministry’s list of tools referred to earlier is MS Excel, again another proprietary tool that might prevent us from sharing our research. Unluckily, many groups, associations, work packages or task forces in open science still use proprietary tools instead of open tools, because it is simply so easy. We need to put much more effort into highly functional, comfortable and interoperable open source tools and then the situation might change.

You might say store it as comma-separated values and, yes, indeed, that helps, but it is not enough. Depending on the version of whatever software you are using to process your data, to visualise your data, things may change. For instance, depending on the number of bits used to store the data in your computer, decimal positions might be cut off or rounded up or down using different algorithms. That’s why data is always closely tied to the very special version of that software you used to generate it, to process it, to visualise it. There is no way to completely separate data from software.

To make things even worse, there is not a single software or tool that we use for a certain research project. We are using more likely a complex IT environment as a matter of course. Most of the components are hidden beneath the surface. There are, for instance, operating systems. I assume you know if you are working on Windows, Linux or Mac, but do you know which drivers, libraries and codecs you have installed? Do you know which execution environment your software is using? Typically not, and to be honest I do not want to know that. Should this not be hidden? However, if you would really draw an image of all the software components you are
using in your research environment, it may look like a wobbly tower of bricks. And I bet that somewhere there is this one tiny little component on which everything else is built – and that makes everything crash if it fails.

Do you remember in 2014 the bug in that openSSL encryption, that we all use for communication on the Internet, that caused a security breach called ‘Heartbleed’ and our local data was put at risk? Do you remember in 2016, there was a bug in the fMRI software that was broadly used to analyse neuroimaging? There was no security reason, but brain areas were declared as active, but in fact they were not. This is a research issue. Do you remember in 2021 the bug in log4j, just a few weeks ago? You could say writing a logbook, is that critical? Yes, it is if that piece of software opens a backdoor to let others enter our computers. Such issues are typical for software that is not well maintained, and this puts our research at risk.

That, however, is enough of the bad news. Let us come to the good news. First, there is a technical approach common in data centres to help tackle such problems. Software, along with the data processed by it, along with the whole computing environment around the software, can be packed into containers, technically boxes. Those containers can be easily moved to other systems, can be easily updated if new versions of any components are released and can be easily recovered after a crash. That is huge progress. You might be reminded of those Jupyter Notebooks. You just put together the data, the code and the documentation and press ‘execute’, and this is possible because of the standardised infrastructure below that software that enables you to easily run and share your research results, including your software.

Using such a container helps not only to maintain but also to publish and reuse the software. Publication comes along with curation, assuring a certain level of quality and, moreover, running software in such a sandbox prevents your system from being damaged.

At the moment, within the German National Research Data Infrastructure my colleague Michael Goedicke and I are preparing such a solution for computer science in Germany. You might compare this approach with the infrastructure we use for electricity, water, the Internet and so on. There is an invisible network of pipelines, cables and switches beneath the surface of our streets, and we just use the outlets in our walls. Plug the device in and everything works, as long as we have the right power adapter in place when we are at a conference oversees. We do not see all the pipelines, cables, and switches. We do not see the many people taking care of it. We just pay our monthly fee and go ahead. This situation can be compared to the IT infrastructure used for research software. Having powerful servers with virtualisation
technology in place and having this container technology in place, your software works. If there is an additional layer to handle your recipes – yes, this is like cooking – you can easily set up different combinations and configurations of software in those containers. This can be used for all the tools we already have in our research portfolio and for all the research software that we will continue to develop.

**Fig. 16: Digital Research Infrastructure**

This sounds difficult, does it not? Do not try to copy this at home. Let the data centre do their jobs. They are professionals for that infrastructure. We are professionals in research. We only have to know how to do our research, not how to maintain such a containerised infrastructure. However, yes, we should finally know how to use such an infrastructure, how to put our software and data in that box and deploy it, or how to maybe adapt what somebody else deployed and use it for our own research. This means additional education and training will be necessary for researchers to achieve these digital skills.

Again, there is good news. There are successful examples, like a series of hackathons carried out by a colleague of mine. Peer Trilcke is one of the leading researchers in digital humanities. He makes use of software to answer questions from philologies through mainly quantitative research. That is why he is also teaching his students how to do that, how to install a computing environment, how to design an algorithm and how to answer their research questions with that. He does it in a very practical way, using textual data, like Drama Corpora, or visual data, like urban graffiti, or audio data from Spotify – cultural data in a broader sense. He creates a kind of data culture from this, and students love that and learn a lot.
Since students are able to learn how to work this way, whatever their research field is, researchers should be able to learn that too. We are currently preparing a systematic evaluation of the competencies that the students gained over the past three years from these hackathons. At the moment, there is such an introductory course established across different fields of study in that discipline, and a broader one is in preparation. I hope it will be available online free for other fields, too, maybe in forms of micro-credentials to easily invoke it. That way future researchers can be trained in using such kind of infrastructure for their later work.

If you were the good fairy and you asked me what I wished for, I would mention the following three things:

• First, recognise the production of research software as a result of research. This affects the time that we need for preparing high-quality code on a project, which requires both funding and certain skills.

• Second, recognise the availability of research software as a valuable outcome. We have to extend our assessment beyond publication count and metrics towards open data and software and we have to consider these, for instance, in our appointment procedures when we select new professors.

• Third, recognise the quality of research software as an important issue. This means that we have to evolve practices for coding towards professional software engineering and to establish effective IT infrastructures and support structures for research software.

This is all about software as a key enabler of research and I think there is no way around it. However, as I told you, software does massively benefit from a proper infrastructure. Is the fairy still in the room? Maybe three more wishes.

• Number four: increase the maturity and integration of current infrastructures and tools. For this we have to identify and to further promote the promising approaches that already exist.

• Number five: adapt the funding instruments to digital infrastructure so we have to add development and maintenance costs to the funding where currently human costs are predominant.

• Finally, number six: establish an international collaboration in infrastructure for research software.

Let us bring together all the relevant stakeholders to support their work. This would bring research significantly forward. And some final good news: we do not have to start from scratch. As we have heard before, there are so many initiatives on the
international level, the European level and the national level involving many smart and dedicated people with many great ideas. Let us please therefore just come together and cooperate, as we are doing here today.

References