Research software at the University of Southampton

November 1, 2019

1 Introduction

Without software, most modern research would not be possible. It is the enabling technology behind major advances, from decoding the human genome to the discovery of the Higgs boson, it lies at the heart of strategically important technologies, such as artificial intelligence, and it is a tool of constant use and fundamental importance in day-to-day research. Despite its pivotal role, support for software and software skills has not kept pace with the staggering speed of the digital revolution, which creates a major barrier that inhibits research and puts results at risk.

A 2014 study [1] found that 92% of UK researchers use software and 69% report that it is fundamental to their research. The near ubiquity of software in research means that it is not possible to disentangle the quality of the software from the quality of the research. Unreliable and untested software leads to unreliable results that cannot be trusted. Trust cannot be established in poorly engineered software because it is too difficult to understand whether the code is reliable. Without trust, researchers will not build on existing software and will instead recreate it themselves: an avoidable duplication of effort and a waste of funding.

In 2019, the Southampton Research Software Group conducted a survey of research staff. The results show that research software is near ubiquitous at the University of Southampton, that it is considered vital to research and that the current provision for supporting software engineering could be considerably improved to benefit both research and enterprise.

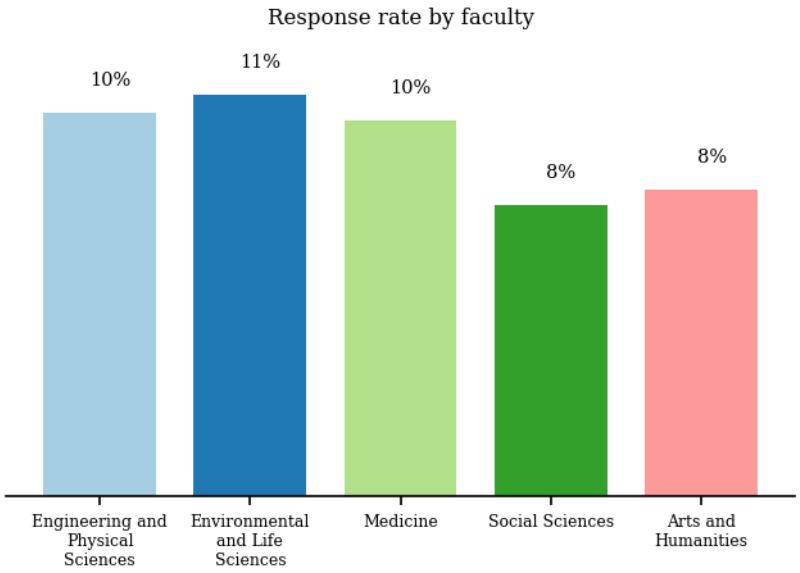
2 Major findings

Of researchers at the University of Southampton:

1. 95% use software in the generation of results they expect to appear in a publication.
2. 73% report software as "vital" to their research.
3. 67% develop their own software, but only 36% of these beleive they can develop reliable software.
4. 78% expected to develop software as part of a funding proposal, but only 26% included costs for software development in the proposal
5. 67% report that hiring a Research Software Engineering from a central pool would be a suitable or perfect way of recruiting software expertise into their group.

2.1 Methodology

The survey was sent to all research staff. In practice this meant sending an email invitation to the mailing lists for all staff employed on an ERE (Education, Research and Enterprise) contract and for all PhD students. The emails were sent on a faculty-by-faculty basis with the following response rate:



In total the survey received 594 usable results that are included in this analysis. It asked 16 questions about the researcher and their use of software. This report provides a brief overview of the results.

Further information about the respondents can be found in the appendices, including the respondents job title and funder, and the number of responses to each question.

2.2 Reproducibility and licences

All resources related to this survey are stored on Github and freely accessible to anyone. The repository [2] contains:

1. A pdf of the original survey
2. The anonymised data collected by the survey
3. The code used to analyse the software
4. This report on the results of the analysis

The above have all been published under an open licence (BSD 3-clause for the code, CC by attribution for the survey and the data).

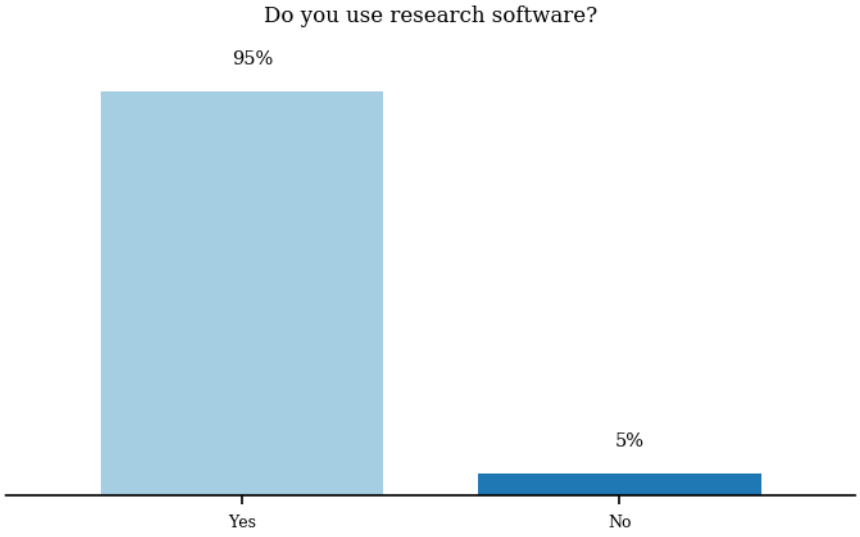
3 What is research software?

"Software" is an umbrella term that disguises a huge amount of complexity and variance. The applications for software are equally complex, from innovations in algorithms to an online service, from the control of a genome sequencer to the analysis of electronic health records, and each is associated with its own particular development requirements.

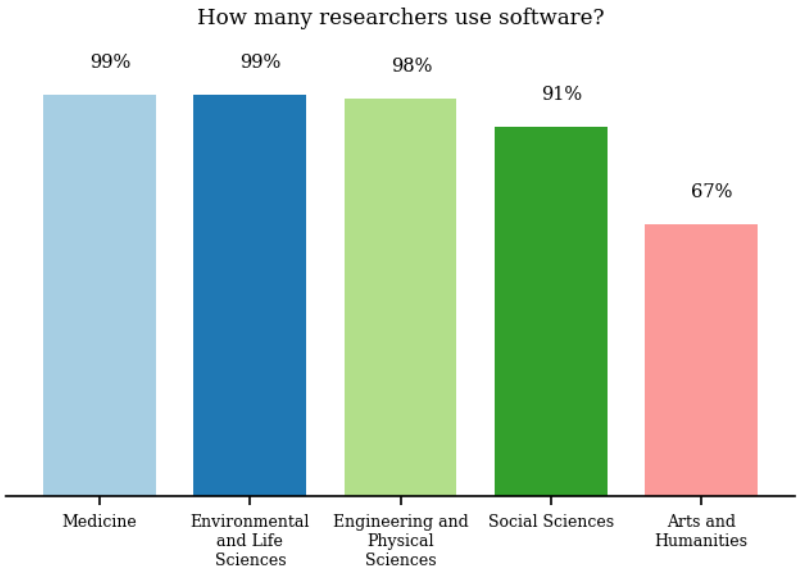
In this survey, we defined "research software" as:

> "any software you have used in the generation of a result that you expect to appear in a publication. This might be anything from a few-line script to clean some data, to a fully fledged software suite. It includes code you have written yourself and code written by someone else."

4 The level of use and importance of software to Southampton researchers

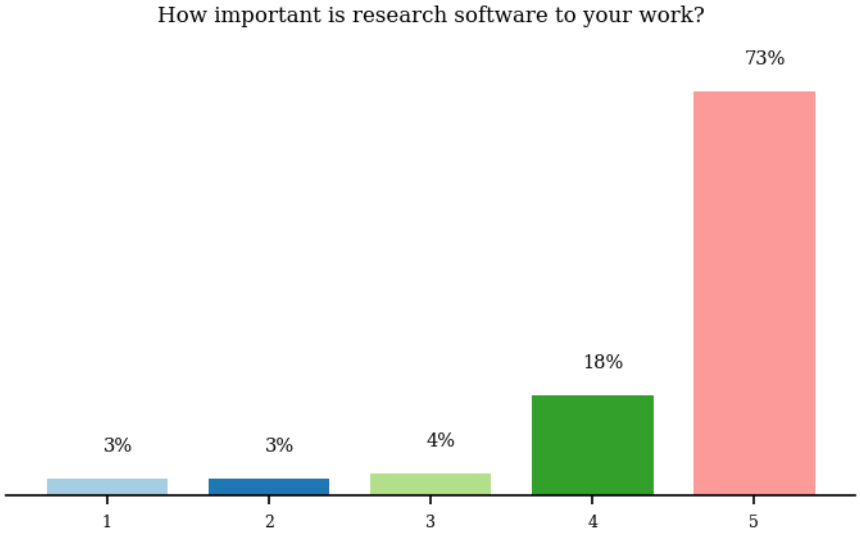


Almost all researchers at the University use research software. Only 5% report that they do not use software, but even this small number seems too large when one consider the near ubiquity of software in research. More insight is gained if we segment the data across faculties.

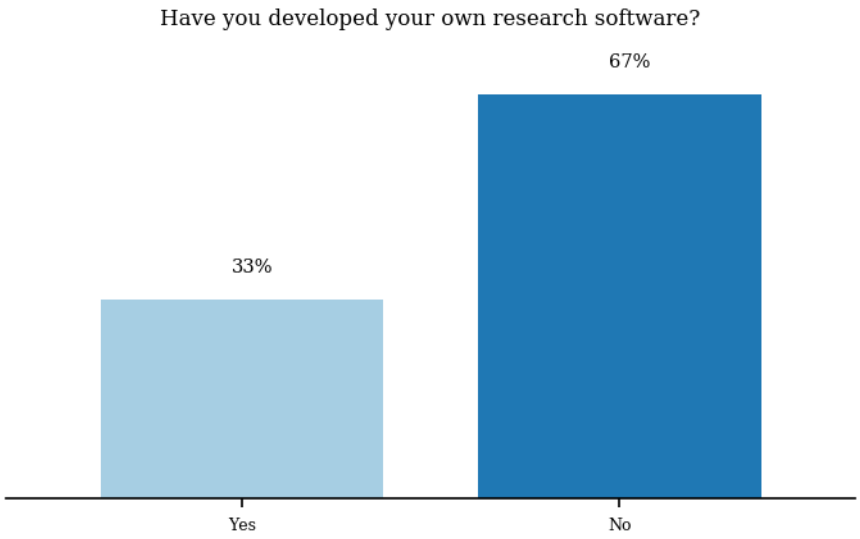


In Medicine, Environmental & Life Sciences, and Engineering & Physical Sciences, the use of software is near 100%. Software use drops slightly in Social Sciences, but it is still above 90%. In Arts and Humanities only 67% report using software in their research, which is understandable in this discipline, although it should be noted that humanities researchers rely heavily on software. The mean percentage of researchers who use software in their research across the university neglecting the contribution from Arts & Humanities is 97%.

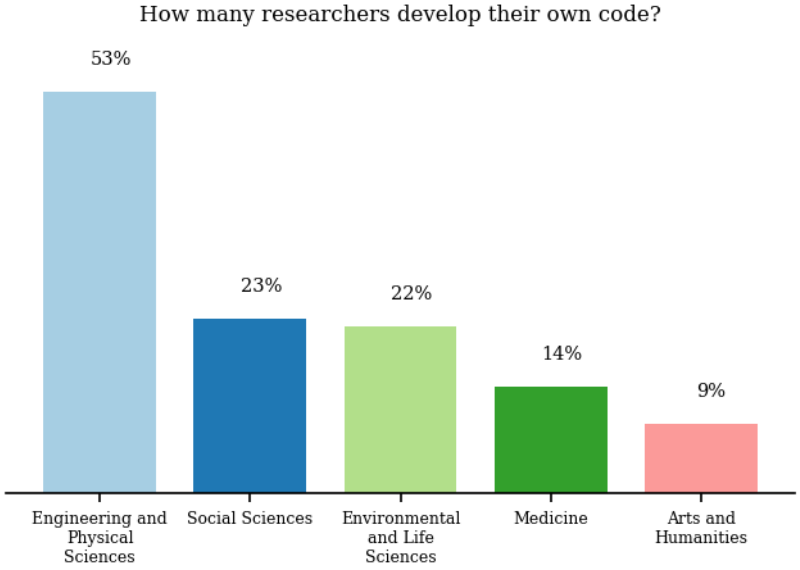
Use of software indicates its importance, but to investigate this issue further we asked researchers to rate how important their research software was to their research (in which a rating of 1 was "not at all" important and a 5 was "vital").



Software is of fundamental importance to the research conducted at the University of Southampton: 95% of researchers use software and 73% state that it is vital to their research.

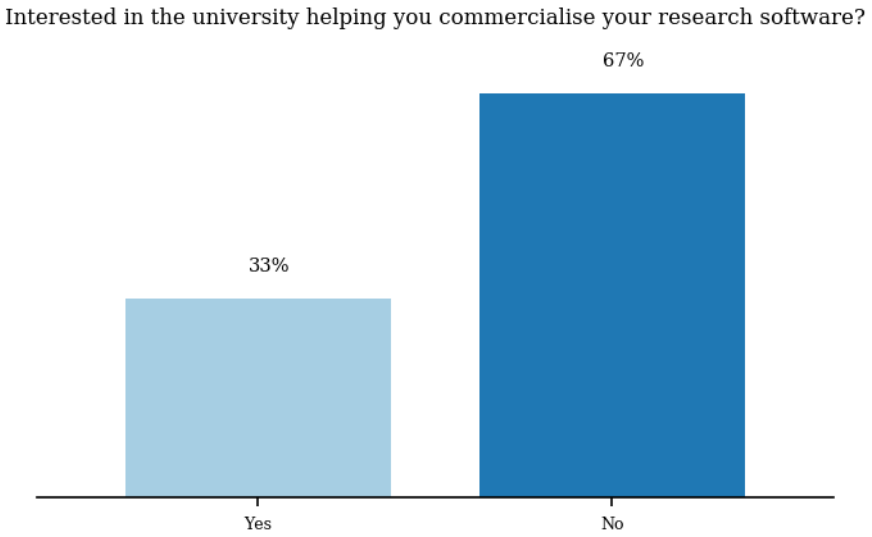


Research tends to create problems that cannot be solved with off-the-shelf software and instead requires the development of bespoke software, which explains why around a third of Southampton researchers develop their own software. The number of researchers who develop their own code varies widely across faculties.

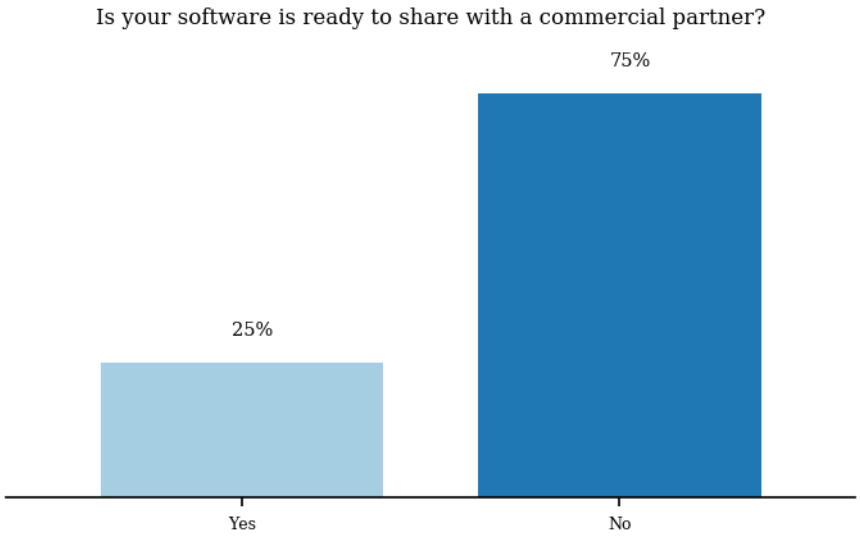


Engineering & Physical Sciences, Environmental & Life Sciences, Medicine and Social Sciences report similar levels of *software use*, but researchers within Engineering & Physical Sciences are far more likely - over double - to write their own code. It seems unlikely that Engineering & Physical Sciences researchers are facing challenges that require significantly more bespoke software development than researchers from other faculties. There lies in this result a huge opportunity: if researchers from the faculties with high software use can be helped to write their own code, they could advance their research.

5 Potential for commercialising software

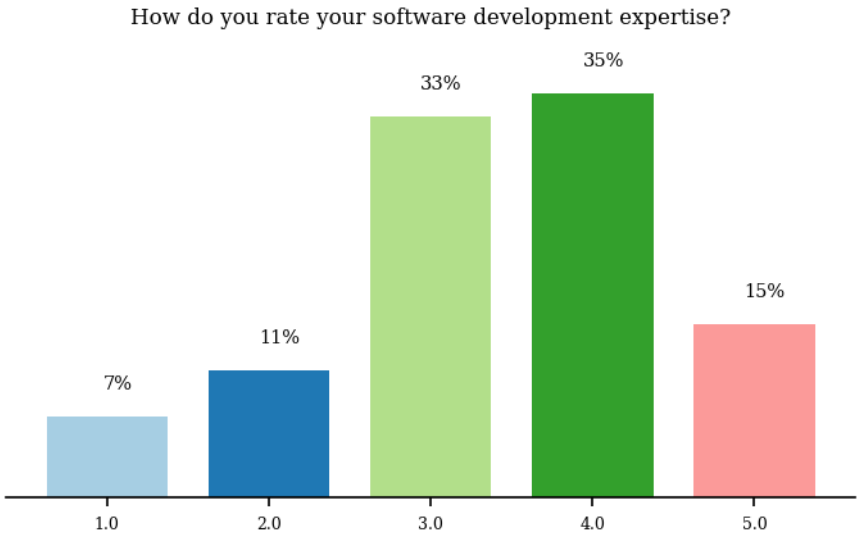


A third of respondents develop their own code, and a third of those are interested in commercialising their software. Extrapolating this result over the University suggests that almost 700 people could have software that they would like to commercialise. With such a large number of potential codes to draw from, there would appear to be significant potential for successful commercialisation.

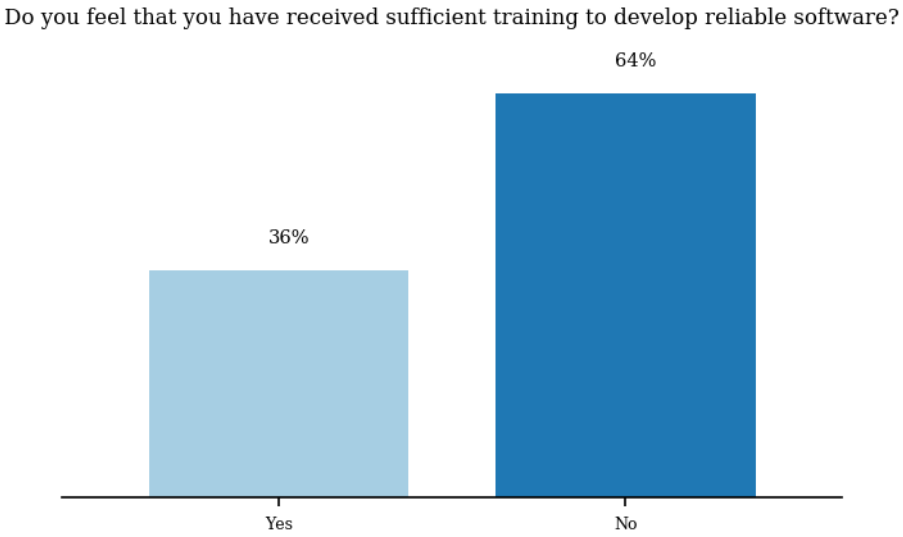


Only 25% of people who wish to commercialise their code would be happy to share it with a commercial partner, which indicates that much code is unfit for commercialisation in its current state. It is likely that 75% of the opportunities to commercialise software developed at University relate to software that lacks robustness or is poorly engineered.

6 Do researchers have access to the software expertise they need?

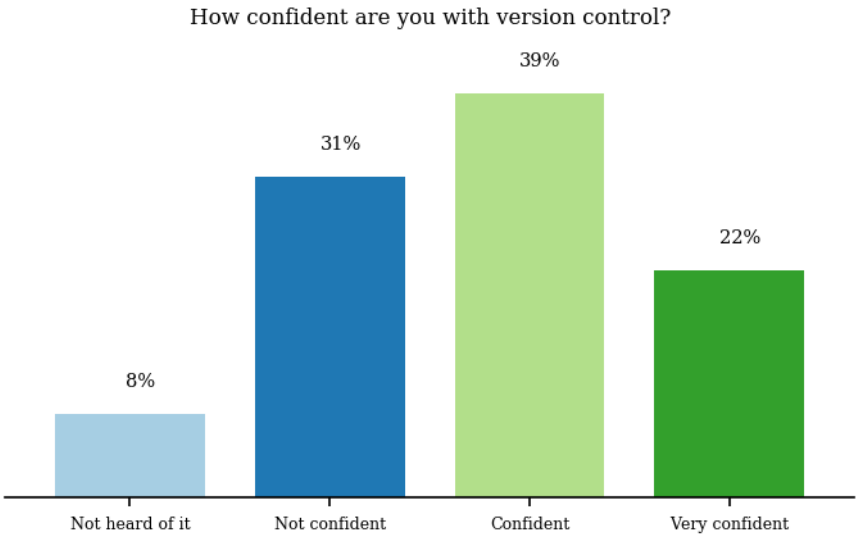


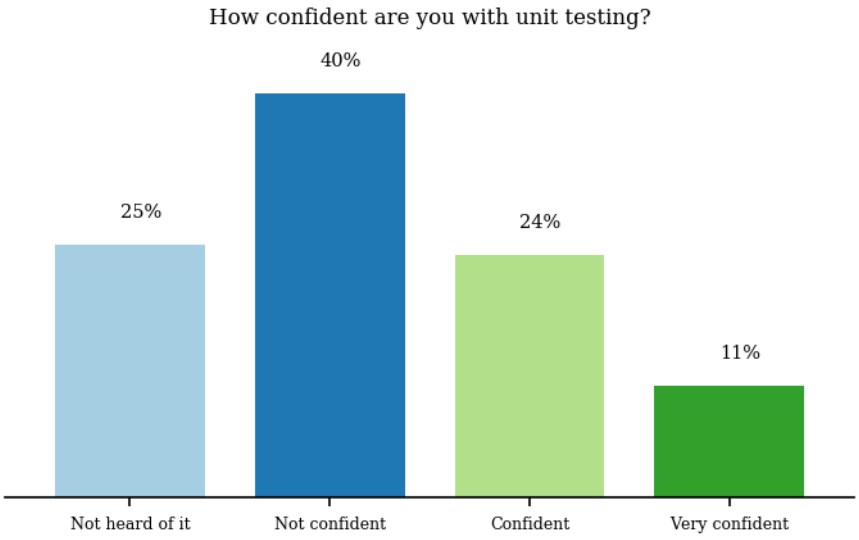
Researchers were asked to judge their software development expertise on a scale from 1 (described as a "beginner") to 5 (described as a "professional"). The largest response (35%) was recorded in the 4 out of 5 category, and 15% described their expertise as professional. Judging one's own expertise can be highly subjective, so we investigated further with questions on training and understanding of a few basic software-engineering techniques.

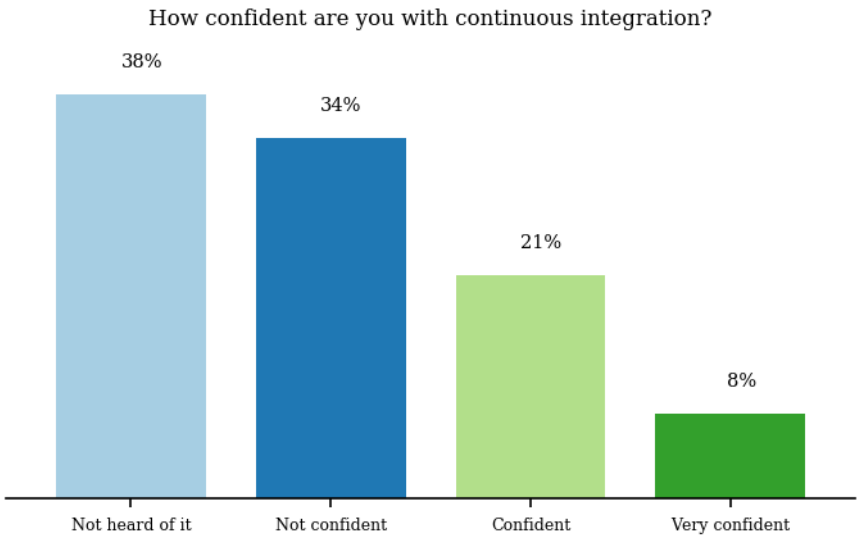


Almost two-thirds of researchers who develop software do not believe that they have received the training needed to produce reliable software. The response to this question obviously raises doubts about the level of development expertise recorded in the previous question. It also suggests that a majority of Southampton researchers are using unreliable bespoke software.

To remove some of the subjectiveness of the development expertise question, we asked further questions about three of the most basic software-engineering techniques: version control, unit testing and continuous integration. Any software engineer would be confident with these techniques.







Almost a tenth of researchers who develop code at the University have not heard of version control - probably the most basic software-engineering technique. This raises some serious concerns about the software developed by these respondents.

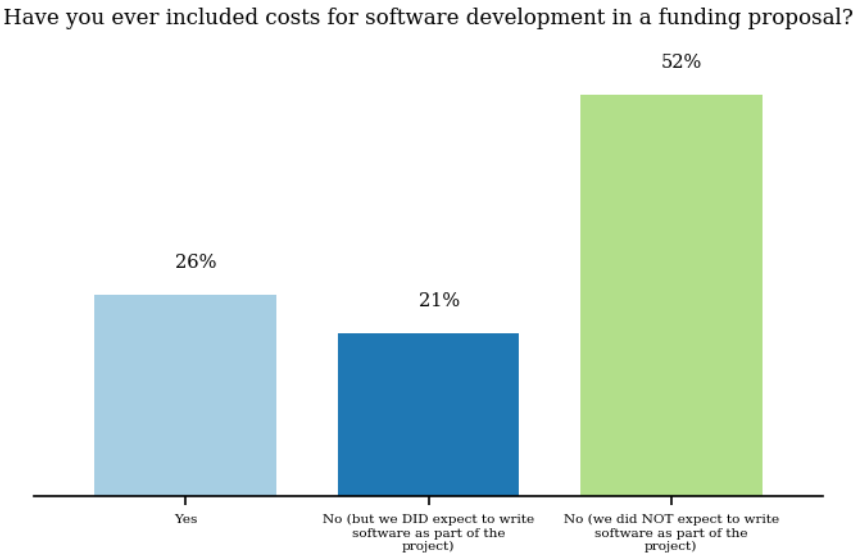
Version control is certainly the most well-known technique, with a majority of researchers reporting that they are confident or very confident with the technique. However, confidence drops with the other two techniques that were assessed, with the majority response to continuous integration question being "Not heard of it".

Good software engineering delivers reliable software, and reliable software delivers trustworthy research. The above results indicate that, despite the reasonably high level of self-assessed software development expertise, a significant number of researchers do not possess basic software-engineering expertise.

7 Access to software-engineering expertise

The modern research group relies on a broad range of skills, many of which researches have to acquire themselves through training - where it exists - or through trial and error. It is simply not possible for all researchers to master of all the skills needed in their group. The obvious solution is to recruit the skills needed from an expert.

We asked whether researchers had included costs for software development in their funding proposals and whether they had hired staff to develop software for them.

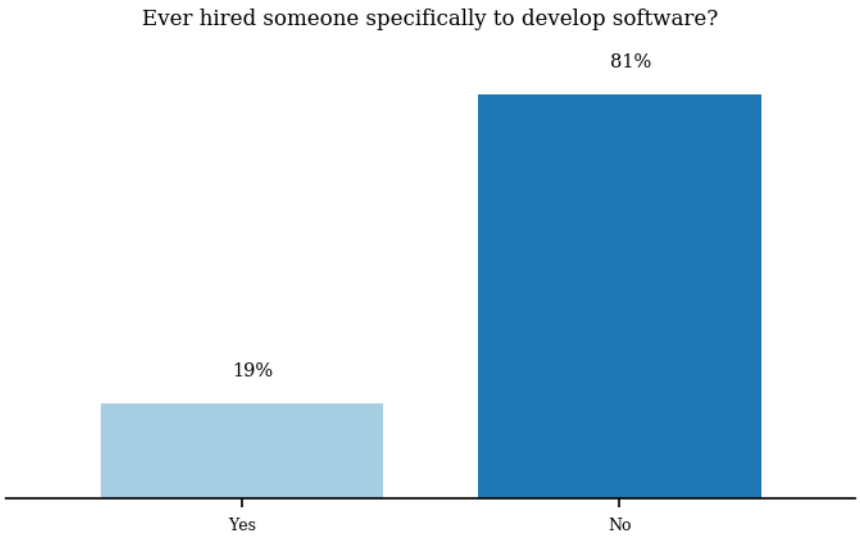


Respondents who were not involved in writing funding proposals were dropped: leaving 280 responses on which to conduct the analysis.

Just under a quarter of respondents did not expect to write software in their research, the rest - almost 80% - expected software to be developed. This reinforces the earlier conclusions on the importance of software to Southampton research.

Over half of Southampton researchers who are responsible for writing funding proposals, expected to develop software but did not include the costs for doing so. This phenomenon does not engender an environment in which researchers are likely to invest the time and judgement needed to develop reliable software. It is most probably caused by the historic belief - demonstrably untrue - that funders do not look favourably on proposals that include costs for software development. It would appear that Southampton researchers require an update on current funder practices.

This result shows that a review of funding proposals will underestimate the amount of software development occuring at the University by around half.

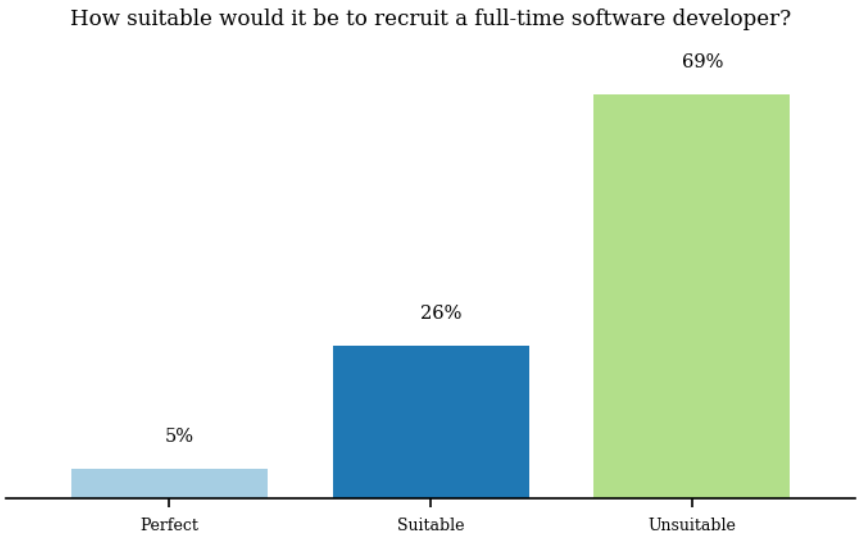


With almost 80% of Southampton researchers expecting to develop software for their research, it seems reasonable to expect a significant number of this group to have hired someone to develop software for their group. However, only 19% of researchers have hired a software developer.

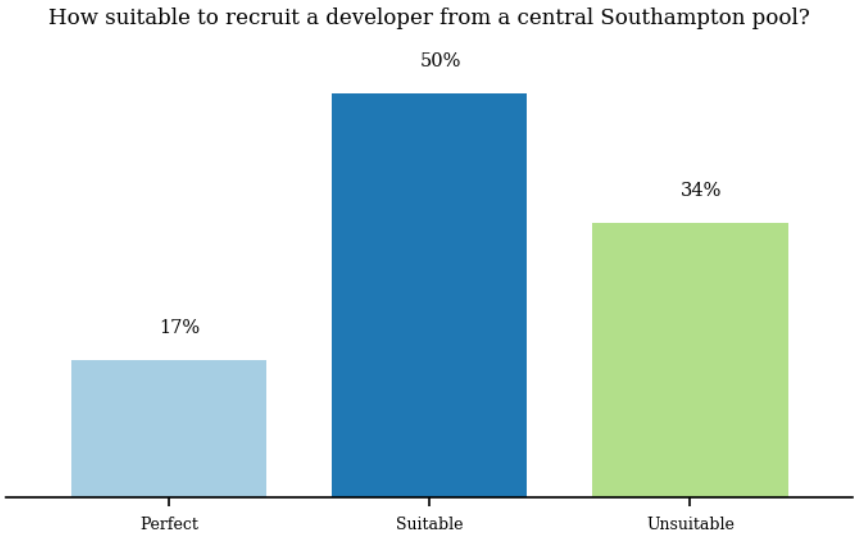
This result indicates that the majority of software development is being conducted by people hired for skills in other areas - most likely PhD students and postdoctoral appointees. There is an obvious problem of handing the responsibility for software development to people who are not trained to develop software.

8 Preferred model for accessing software expertise

There is significant demand for software engineering expertise at the University. We asked respondents for their preferred model for recruiting this expertise.



Over two-thirds of researchers would find employing a full-time software developer unsuitable. This is most likely because they lack either the funds to hire a full-time developer or do not generate enough software development work to keep one employed.



The alternative option within the University of Southampton (and the other 27 UK universities with a Research Software Engineering group) is to hire a Research Software Engineer to work on a project. This model has many benefits: the researcher gains access to professional software engineering expertise, they pay only for the time they need, the researcher does not have to conduct recruitment or concern themselves with the line management of a position they may not have the expertise to understand, and the cost for is similar to that of any other research position.

Based on these benefits, it is unsurprising that over two thirds of Southampton researchers find this model for recruiting expertise either suitable or perfect.

9 References

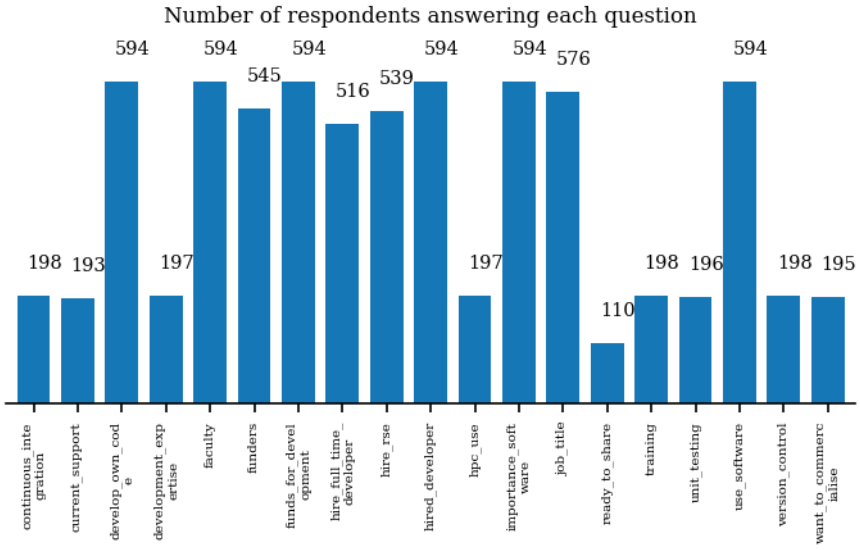
1. National software survey: "S.J. Hettrick, et al, UK Research Software Survey 2014", DOI:10.5281/zenodo.1183562.
2. Github repository for this project: https://github.com/Southampton-RSG/soton_software_survey_analysis_2019

10 Appendices

We collected information on the number of responses to each question and asked further questions in the survey to understand the demographics of the respondents, their use of HPC and their rating for software support.

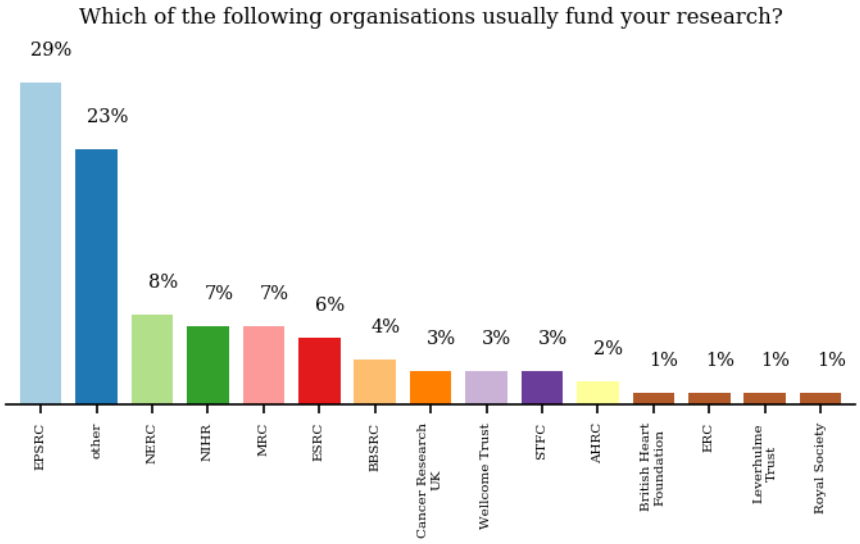
10.1 Appendix A

Number of respondents to each question:



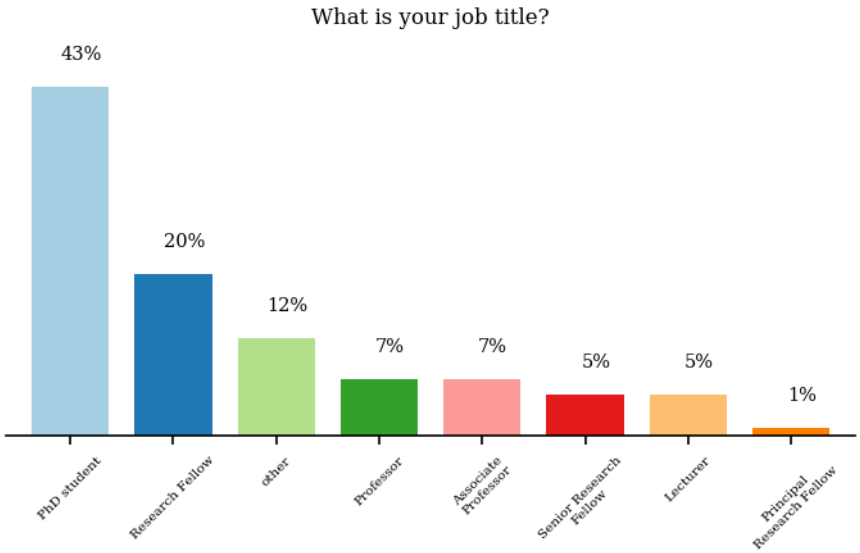
10.2 Appendix B

Origin of research funding of respondents:



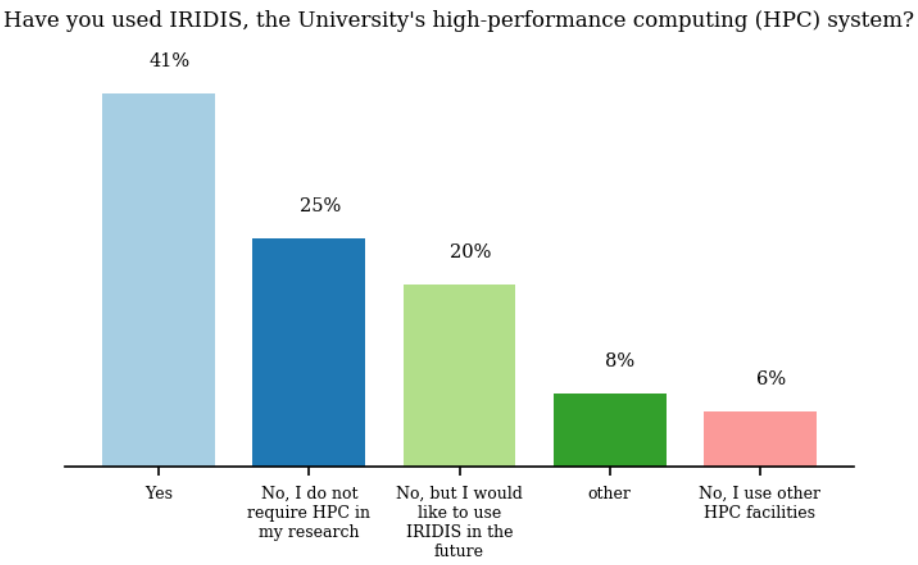
10.3 Appendix C

Job title of respondents:



10.4 Appendix D

We investigated whether researchers had ever used the University's HPC system:



10.5 Appendix E

Rating of current support for software: