Research software at the University of Southampton

December 10, 2019

A report by the Southampton Research Software Group: Brown A., Crouch S., Graham J., Grylls P.J., Hettrick S.J., Mangham S.W., Robinson J. & Wyatt C.

1 Introduction

Most research would not be possible without software. 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 within academia has not kept pace with the rapid adoption of software as a research tool. This is 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. It is not possible to disentangle the reliability of the software used to generate results from the reliability of the research itself. Unreliable 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 software 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 resources.

In 2019, the Southampton Research Software Group (SRSG) [2] conducted a survey of research staff to identify the relationship between software and research at the University of Southampton. The SRSG works across faculties to provide software engineering expertise and training to researchers. The survey found that software use is near ubiquitous across all but one faculty, researchers report that software is vital to their work, a third of researchers develop their own software, and around a third who develop software are interested in commercialisation. Worryingly, around two thirds of staff who develop software do not believe they have had sufficient training to develop reliable software.

Software has an almost unparalleled power to advance research and enterprise. Much of this potential is likely to remain unrealised at the University, unless researchers are given access to the necessary software skills and expertise.

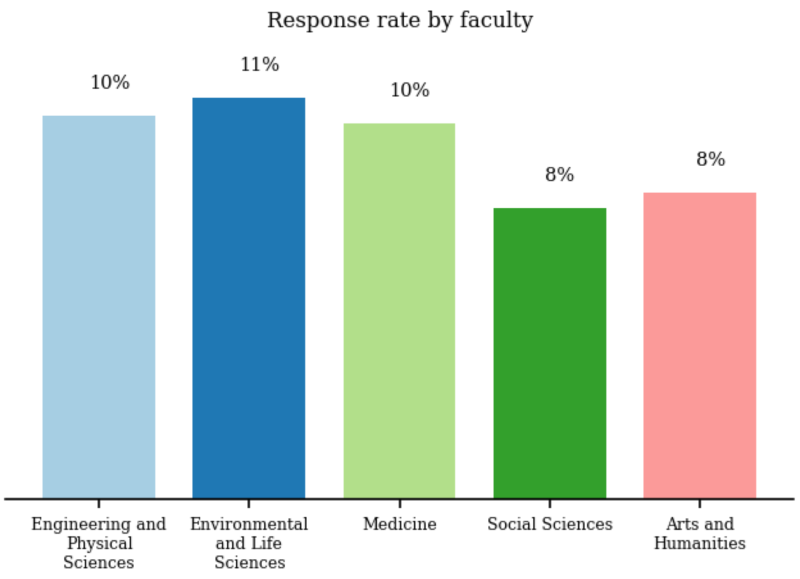
2 Major findings

- 95% of researchers use software to generate results they expect to appear in a publication.

- 73% report software as “vital” to their research.
- 33% develop their own software and, worryingly, of these software developers, 64% do not believe they have received sufficient training to develop reliable software.
- 47% expected to develop software as part of a funding proposal but, despite this fact, less than half of these included costs for software development in the proposal.
- 67% report that hiring a Research Software Engineer from a central pool would be either a perfect or suitable way for providing access to the skills and expertise they require.

2.1 Methodology

The survey was delivered to all research staff via an email invitation sent to the mailing lists for all staff employed on an ERE contract (Education, Research and Enterprise) and all PhD students. To counter response bias, a prize draw was organised for respondents, the survey was kept short, and recipients of the email were informed of the importance in responding even if they did not use software. The emails were sent on a faculty-by-faculty basis with the following response rate:



The survey asked 16 questions about the researcher and their use of software. It received 594 usable results which are included in this analysis.

The appendices to this report include information about the number of responses to each question (not all were mandatory), and further information about the respondents such as the respondents' job titles and funders.

2.2 Reproducibility, licences and citation

All resources related to this survey are stored on Github and freely accessible to anyone. The repository [3] 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).

2.2.1 Citation

The results, analysis and report are published under the DOI: 10.5281/zenodo.3569549

If you reproduce any of the results or text from this study, please cite:

- Brown A. (ORCID: 0000-0003-2069-0953), Crouch S. (ORCID: 0000-0001-8985-6814), Graham J., Grylls P.J. (ORCID: 0000-0001-9677-5852), Hettrick S.J. (ORCID: 0000-0002-6809-5195), Mangham S.W. (ORCID: 0000-0001-7511-5652), Robinson J. & Wyatt C. "Research software at the University of Southampton". 10 December 2019.

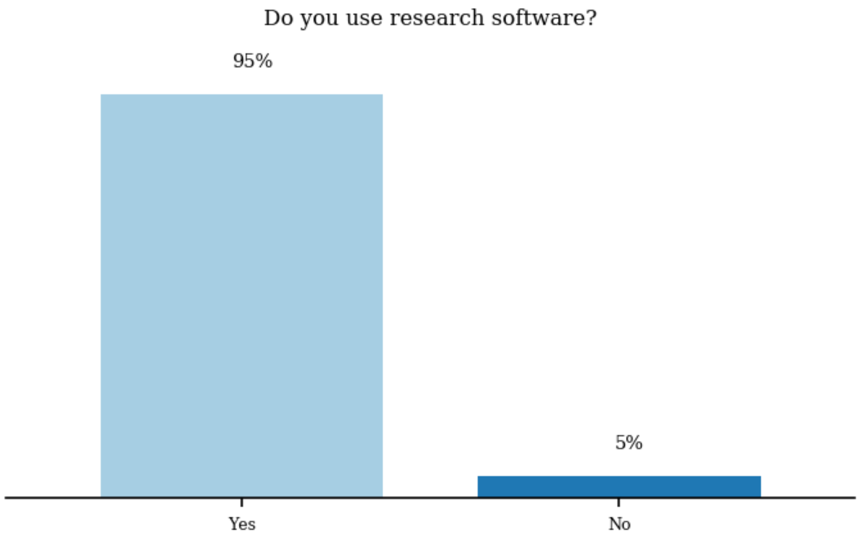
3 What is research software?

"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:

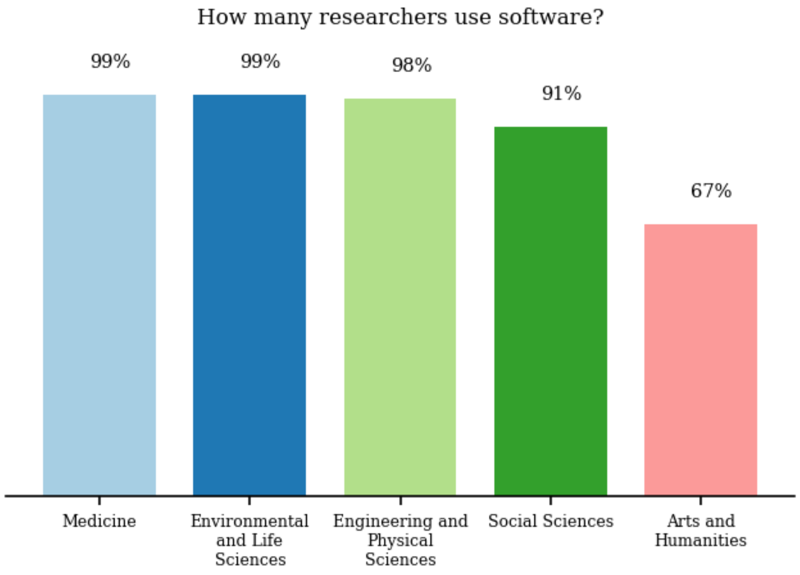
> "any software you have used in the generation of a result that you expect to appear in a publication. This might be anything from a few-line script to clean some data, to a fully fledged software suite. It includes code you have written yourself and code written by someone else."

Our primary concern is the reliability and reproducability of research conducted at the University of Southampton, so we chose to define *software* based on the role it plays in generating results.

4 Use of software and importance to research

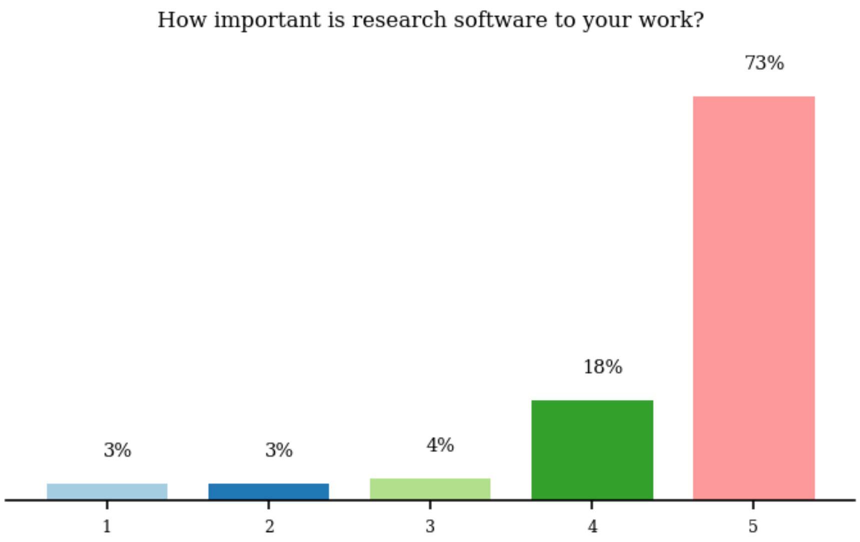


The vast majority of researchers at the University use research software. Only 5% report that they do not, but even this number seems too large. More insight is gained if we segment the data across faculties.

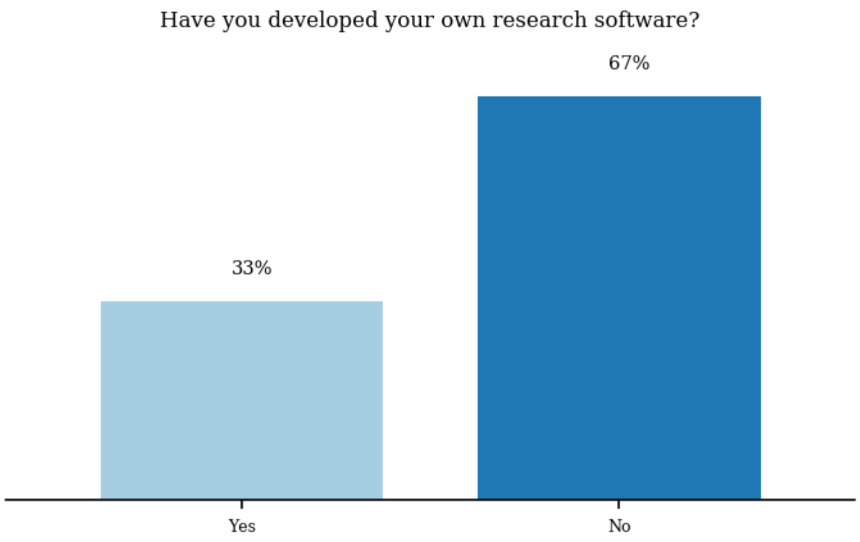


In Engineering & Physical Sciences, Environmental & Life Sciences and Medicine, the use of software is near 100%. Software use drops slightly in Social Sciences - but it is still above 90%. In Arts and Humanities only 67% report using software in their research, which is understandable in this discipline, although it is 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%.

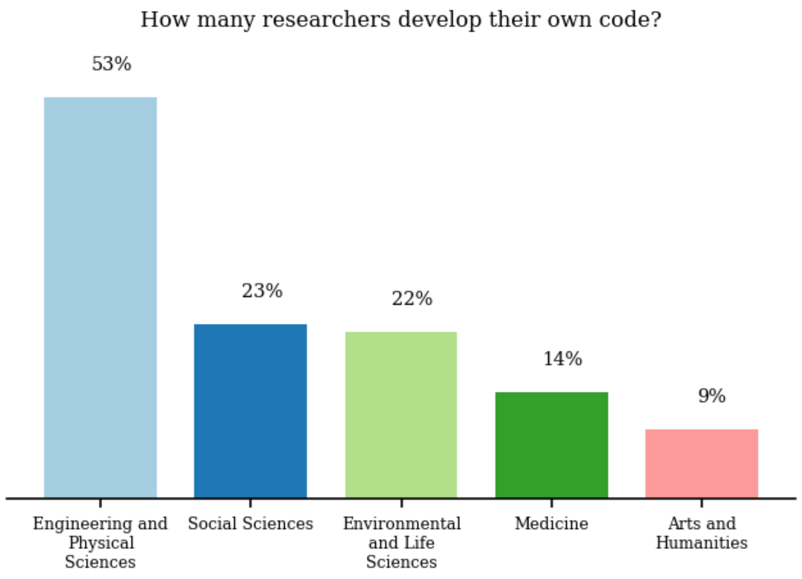
Use of software provides insight into its prevalence, but only hints at its importance to research. To investigate this issue further we asked researchers to rate how important research software was to their research (in which a rating of 1 was “not at all” important and a 5 was “vital”).



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.



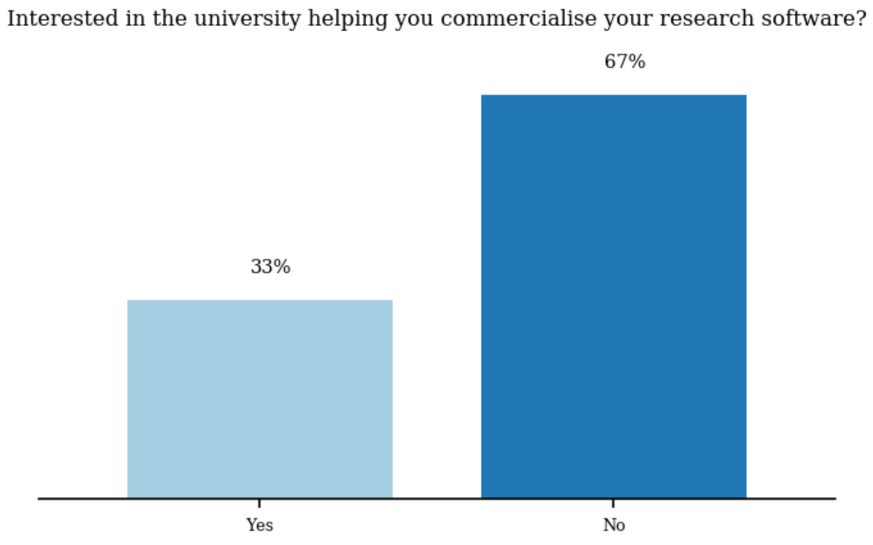
Around a third of researchers at the University develop their own software. This is not surprising: the specific problems faced in research tend to require bespoke solutions. However, the number of researchers who develop their own software varies widely across faculties.



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 all other faculties. It is a generally held opinion that researchers from EPS background are more likely to have experienced training in programming at some point in their education. However, as we will see later in regards to training, this does not appear to be the case.

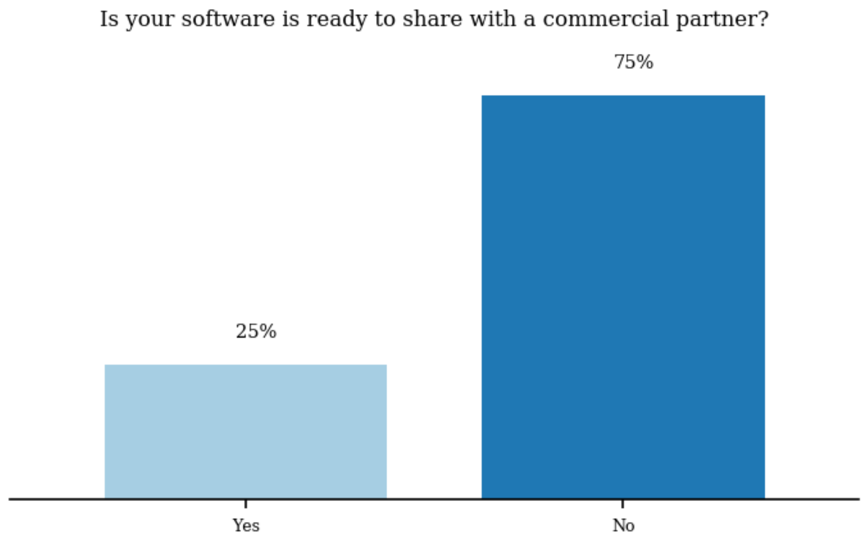
Whatever the cause, there lies in this result a huge opportunity: training researchers from faculties that present high software use but low software development could have a significant effect on the speed, scale and reliability of their research.

5 Potential for commercialising software



A third of respondents who develop their own code 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 an abundance of software to draw from, there would appear to be significant potential for successful commercialisation of some of it.

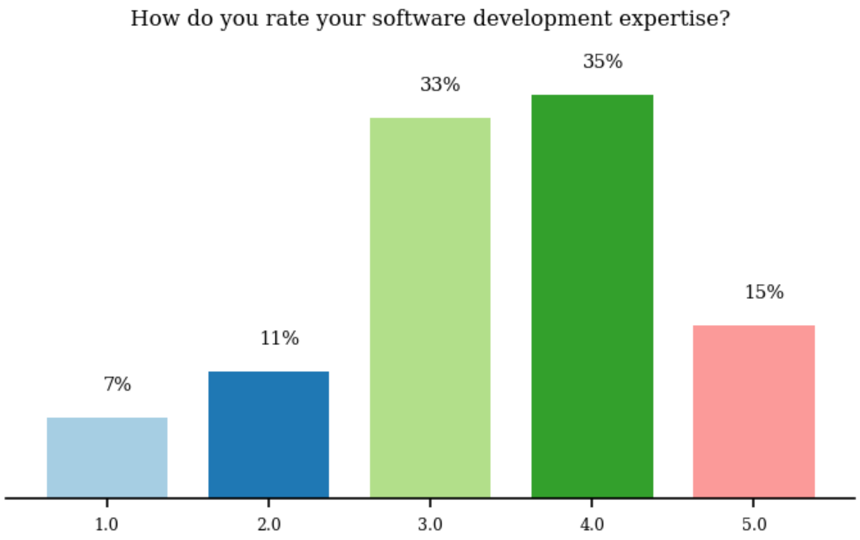
It requires more than just a good idea to produce commercial software, the software itself must stand up to the demands of its customers. To investigate the readiness of the software, we asked researchers who develop software whether it was ready to share with a commercial partner.



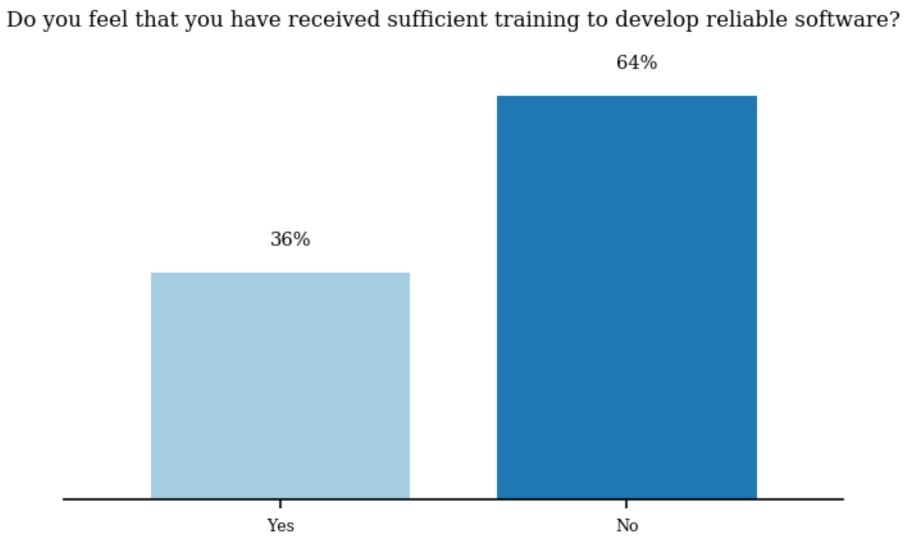
75% of software is not ready to share with a commercial partner, and this has to be viewed as a strong indicator that the software is unfit for commercialisation in its current state.

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

Producing reliable software, whether to use alone or share with research or commercial partners, requires access to software expertise. We asked researchers to evaluate their skills with software development.



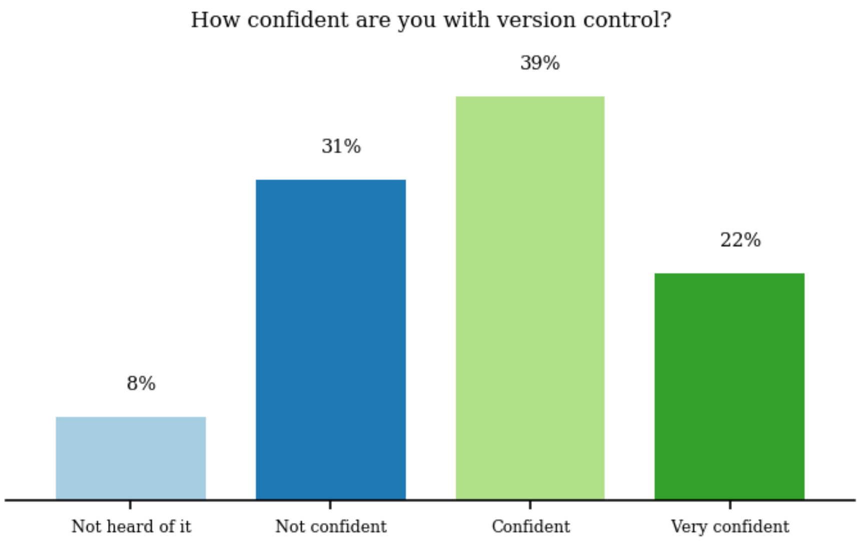
Researchers who develop software were asked to judge their software-development expertise on a scale from 1 (described as a *beginner*) to 5 (described as a *professional*). The majority response (35%) was recorded in the 4 out of 5 category. 15% of staff described their expertise as *professional*, and only 18% judged themselves as having poorer than average software-development expertise. At first sight, this result is grounds for some optimism. However, judging one's own expertise can be highly subjective. We investigated expertise levels further with more objective questions on training and on understanding of a few basic software-engineering techniques.

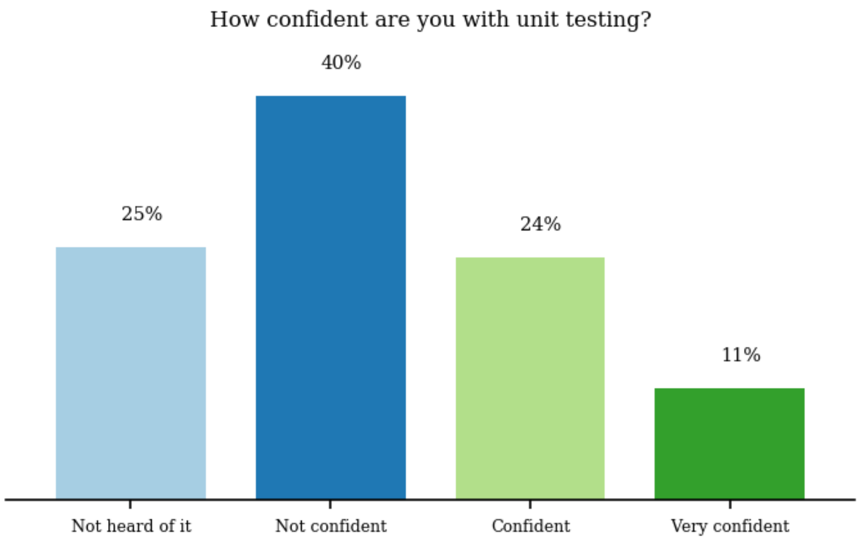


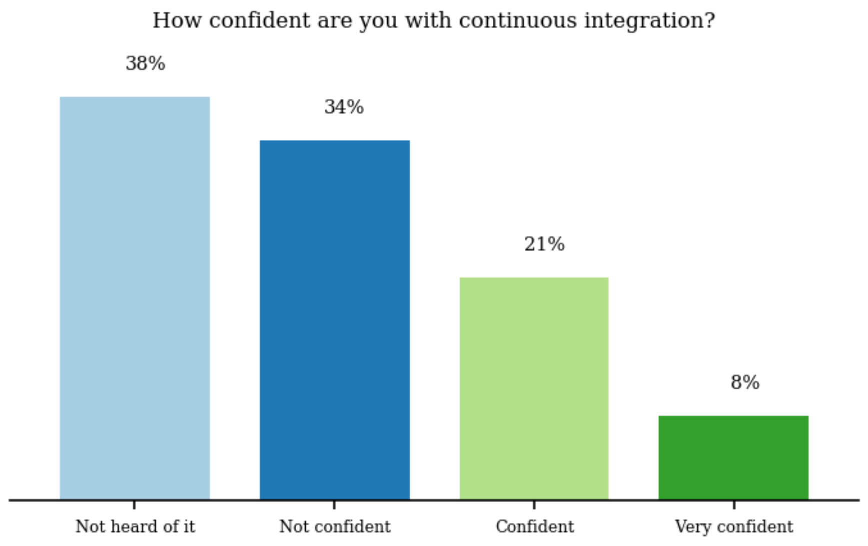
Almost two-thirds of researchers who develop software do not believe that they have received the training needed to produce reliable software. This raises concerns about the publications which are based on results generated by homegrown software.

The response to this question also raises doubts about the accuracy of the self-evaluation of development expertise recorded in the previous question: a third of researchers who evaluated their development expertise as *professional* reported that they had not received sufficient training to develop reliable software.

What is judged as professional development expertise will vary widely dependent on the experience of the person answering the question. To further investigate expertise we focussed on more objective questions about three basic software-engineering techniques: version control, unit testing and continuous integration. Any software engineer would be confident with these techniques.

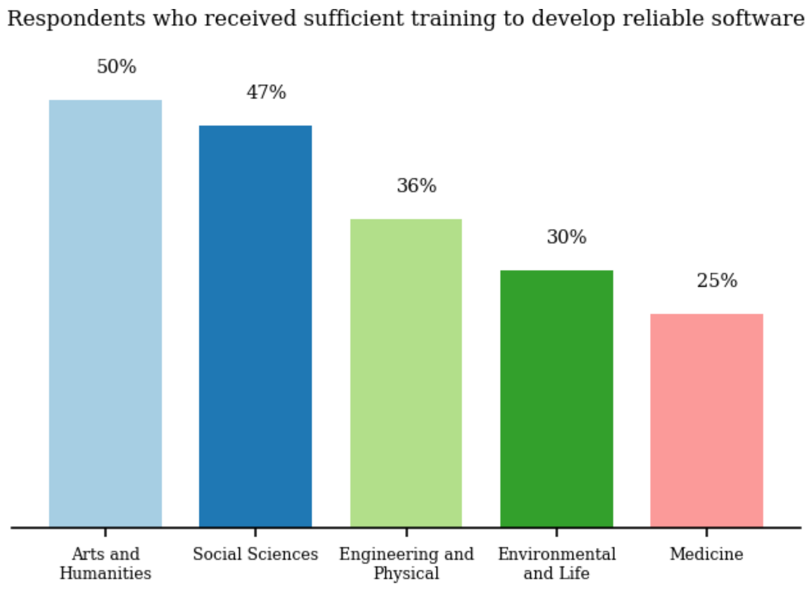






Version control is certainly the most well-known technique, with a majority of researchers reporting that they are confident or very confident with the technique. Yet one in ten of researchers who develop code have not heard of this most basic software-engineering technique. As we move to more advanced (yet still basic) tools, confidence levels drop with only a quarter of respondents confident with unit testing and the majority not having even heard of continuous integration.

In conclusion, despite the reasonably high level of self-assessed software development expertise, it seems reasonable to state that a significant number of researchers who develop software lack expertise in software engineering and this is likely to lead to the development of unreliable software.



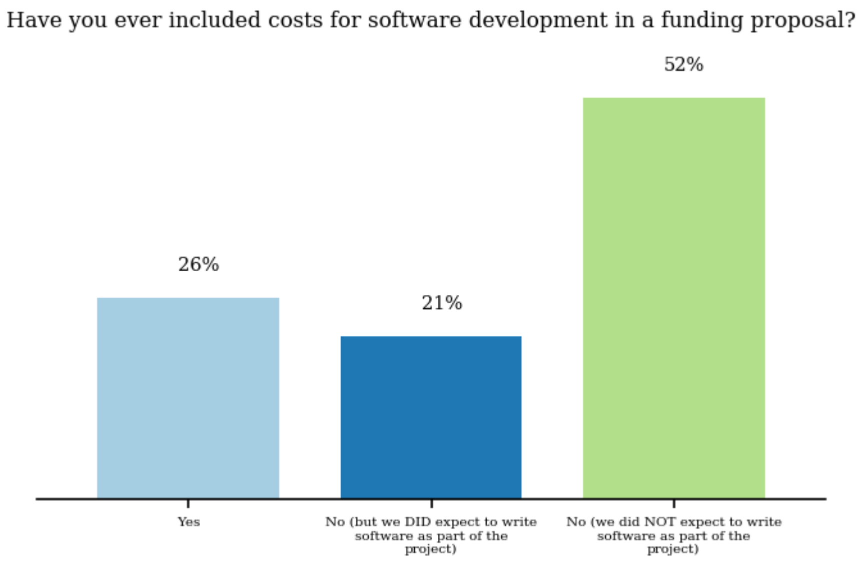
As discussed above, the percentage of researchers who develop their own code in Engineering & Physical Sciences is more than double that of other faculties. A possible reason for this large difference might be that EPS researchers receive more training in software development. To investigate this theory, we took the question on whether researchers had received sufficient training to develop reliable code and segmented it by faculty. The chart above shows the percentage of researchers from each faculty who believed that they *had* received sufficient training to develop reliable code.

The response from Arts & Humanities can be discounted: after the segmentation on developing code, this left only 4 respondents and this cannot be taken as representative of the faculty as a whole.

The greatest percentage of respondents who believe they have received sufficient training to develop reliable software are based at the Faculty of Social Sciences (47%). The faculties of Environmental & Life Sciences and Medicine display lower percentages (30% and 25% respectively) than Engineering & Physical Sciences (36%), but the difference is not significant. It would appear that researchers within faculties other than Engineering & Physical Sciences are not held back from developing their own software by their perception of their ability to develop reliable software.

7 Access to software-engineering expertise

Research groups rely on a broad range of skills which researchers are expected to acquire through training - where it exists - or through trial and error. But with modern research relying on an ever wider range of skills, it is simply not possible for all researchers to master all of the skills their research requires. The obvious solution is to recruit skills from an expert. To investigate this phenomenon specifically in the field of software, 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.

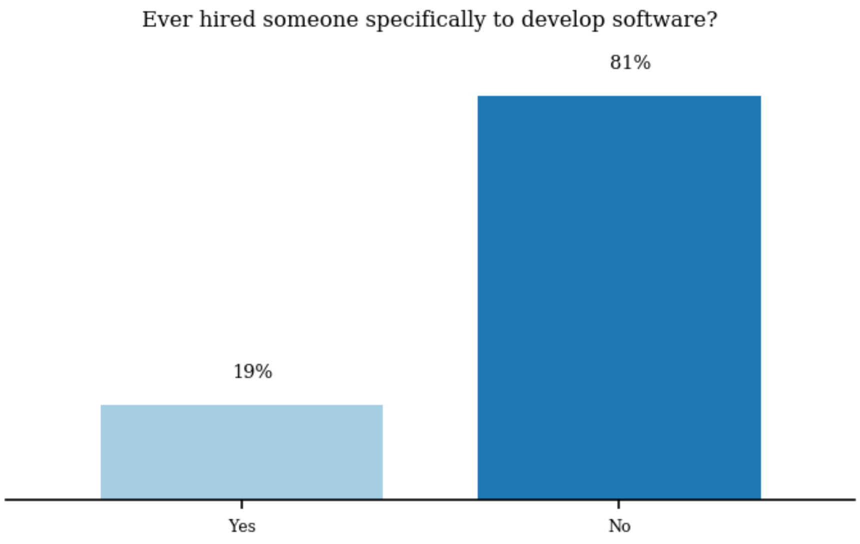


Respondents who were not involved in writing funding proposals were dropped: leaving 280 responses on which to conduct the analysis.

Just over a half of respondents did not expect to write software in their research, which is in conflict with the response to the earlier question which found that 67% of researchers develop their own software.

47% expected to develop software as part of a funding proposal, but less than half of these included costs for software development in the proposal. It cannot be good practice to expect work to be completed that has not been budgeted for. Further investigation is required to understand why people would not include costs in a bid when they knew those costs would be incurred. It is likely that postgraduate students and postdoctoral researchers would be expected to pick up any software development that is required. It may also be related to the incorrect belief that funders do not look kindly on software-development costs in research bids.

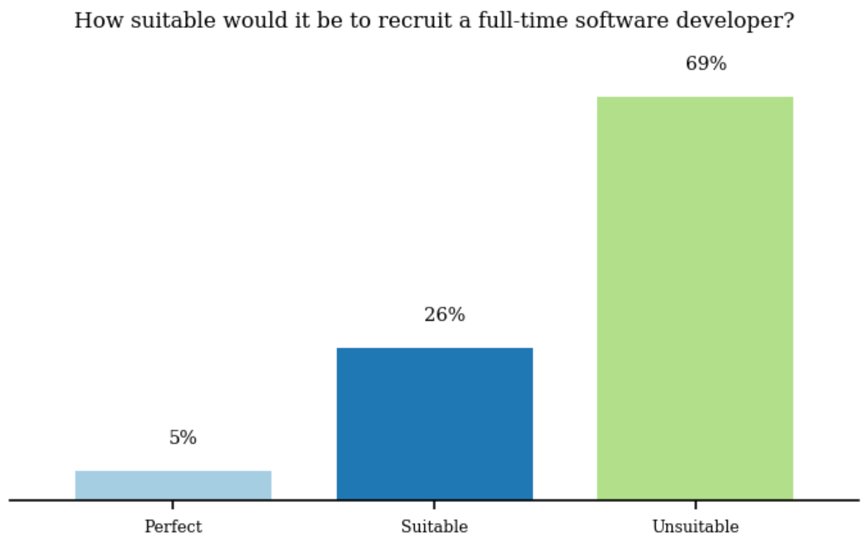
This result also shows that a review of bids won by the University will underestimate the amount of software development occurring at the University by around a half.



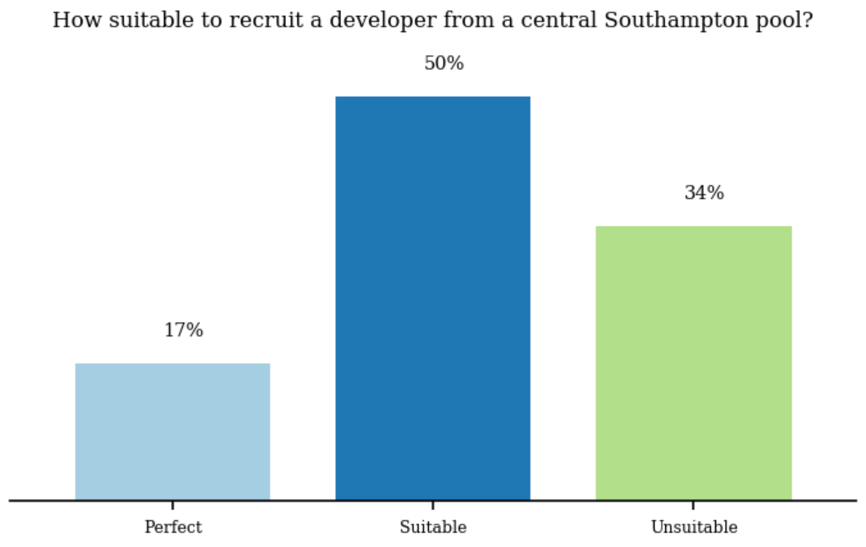
With almost half of researchers at the University expecting to develop software in their proposals, it seems reasonable to expect a significant number will have hired a software expert to conduct this development. However, only 19% of researchers have hired someone specifically to develop software. This indicates that the majority of software development is being conducted by people hired for skills in other areas - most likely postgraduate students and postdoctoral researchers as discussed above. 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

We expected to find a significant demand for software engineering expertise at the University, so 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 for their needs. This is not a surprising result. There are significant difficulties in finding the funds and the work to keep a full-time developer employed within all but the largest research groups. What's more, it's very difficult to design a recruitment strategy and to interview software experts if you lack software expertise in your group and therefore do not understand how to select the good software experts from the bad.



The alternative option within the University of Southampton (and the other 27 UK universities with a Research Software Engineering group [4]) is to hire a Research Software Engineer from the Southampton Research Software Group [2]. 67% of researchers would find this solution either perfect or suitable to meet their development needs.

9 Conclusion

The University of Southampton follows the same software use patterns as found in the national software survey [1] and studies conducted by other groups around the world. Use of software is near ubiquitous across disciplines, and the software plays a vital role in the conduct of research.

Around a third of researchers develop their own software, but there is significantly more development occurring in Engineering and Physical Sciences despite software use in that faculty being similar to the other faculties. The cause for this imbalance is not known, but helping more faculties develop their own software could present a major opportunity for increasing the speed, scale and reliability of their research.

There is a significant interest in commercialising software, but much of the potential is limited by software that is not yet ready to be shared with commercial partners. This indicates a lack of access to skills and the time needed to conduct software development.

Although researchers who develop their own software rate their software-development expertise quite highly, almost two-thirds do not believe that they have received sufficient training to develop reliable software. Objective questions based around software-engineering expertise indicate that few researchers possess professional expertise with software. The combination of near-ubiquitous software use, vital software importance and low software-development expertise is an obvious area of concern for the University.

Despite almost half of researchers expecting to develop software when writing their funding proposals, only around half included costs for software development in the proposal and only 19% have recruited a software expert specifically for this role. This indicates that much software is developed at the University by people whose main area of expertise is not software development.

69% of researchers stated that hiring a full-time software developer would be unsuitable for their needs. This is likely due to difficulties in recruitment and in providing sufficient funds and work to retain a full-time developer. The Southampton Research Software Group, like the RSE groups that exist in 27 other universities across the UK, allow researchers to hire a Research Software Engineer when they need one. The researcher gains access to a range of professional software engineering expertise that very few, if any, research groups possess, and they only pay for this expertise when it is needed. Researchers need not concern themselves with recruitment or line management of the Research Software Engineer. Based on these benefits, it is unsurprising that nearly 70% of researchers find the RSE model for recruiting expertise either perfect or suitable.

10 References

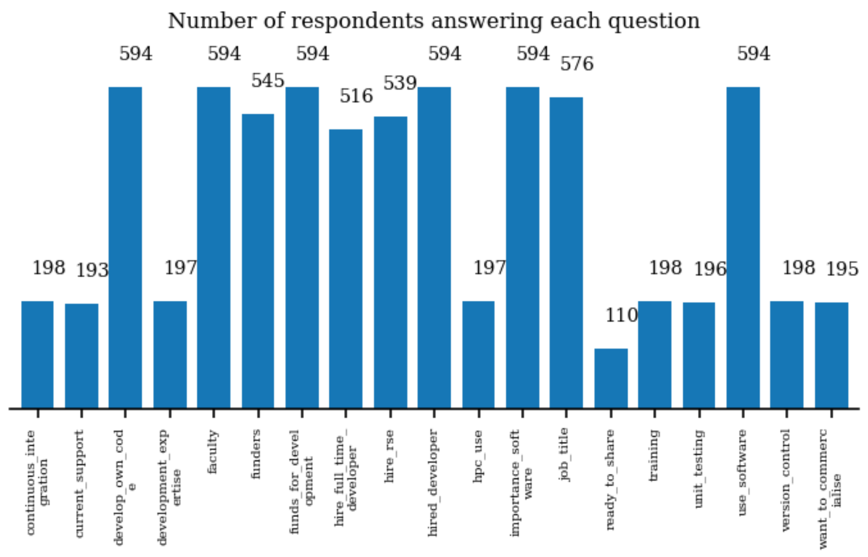
1. National software survey: "S.J. Hettrick, et al, UK Research Software Survey 2014", DOI:10.5281/zenodo.1183562.
2. https://rsg.southampton.ac.uk
3. Github repository for this project: https://github.com/Southampton-RSG/soton_software_survey_analysis_2019
4. https://society-rse.org/community/rse-groups/

11 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.

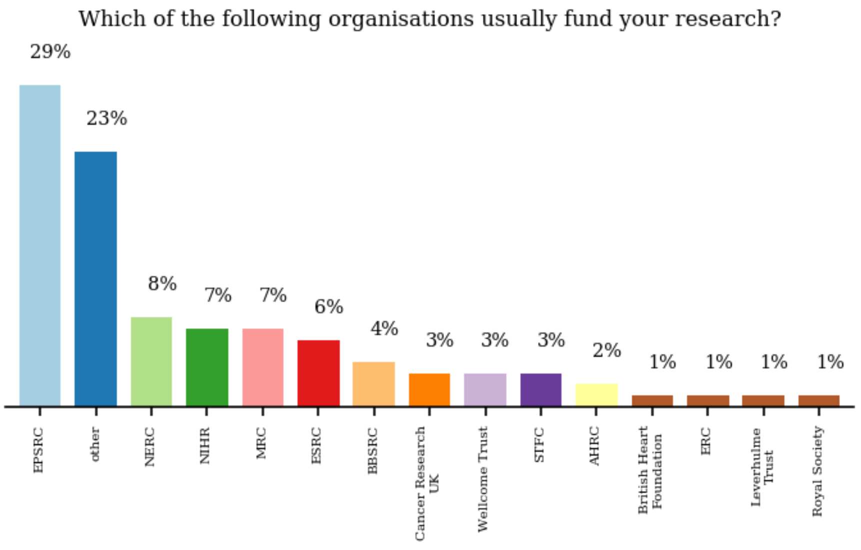
11.1 Appendix A

Number of responses to each question:



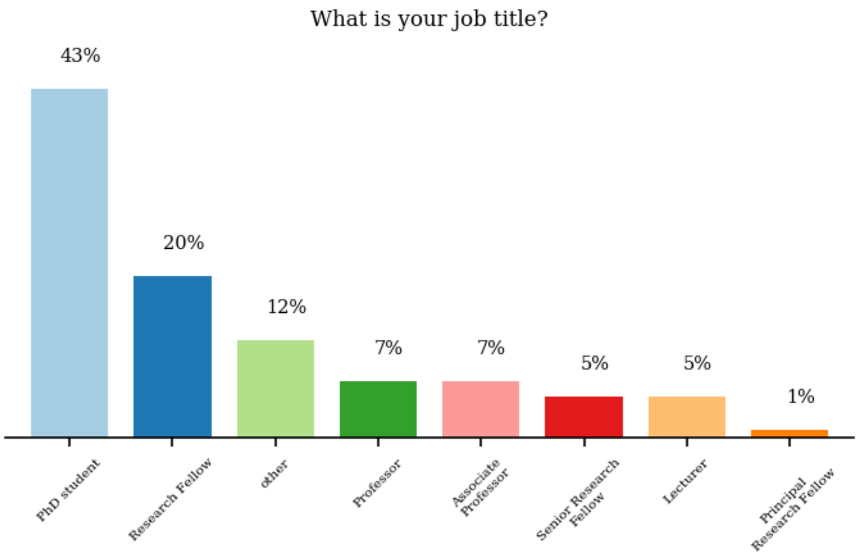
11.2 Appendix B

Origin of research funding of respondents:



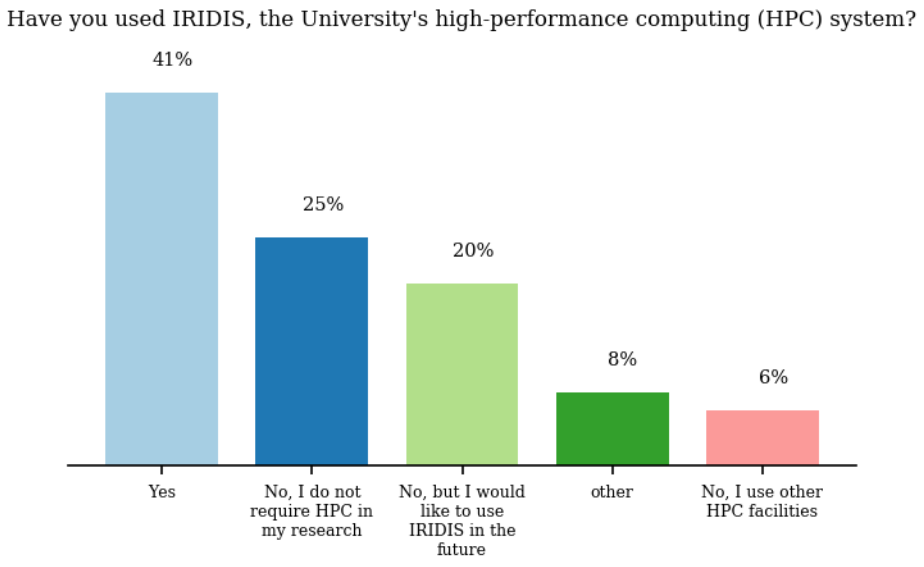
11.3 Appendix C

Cleaned job title of respondents:



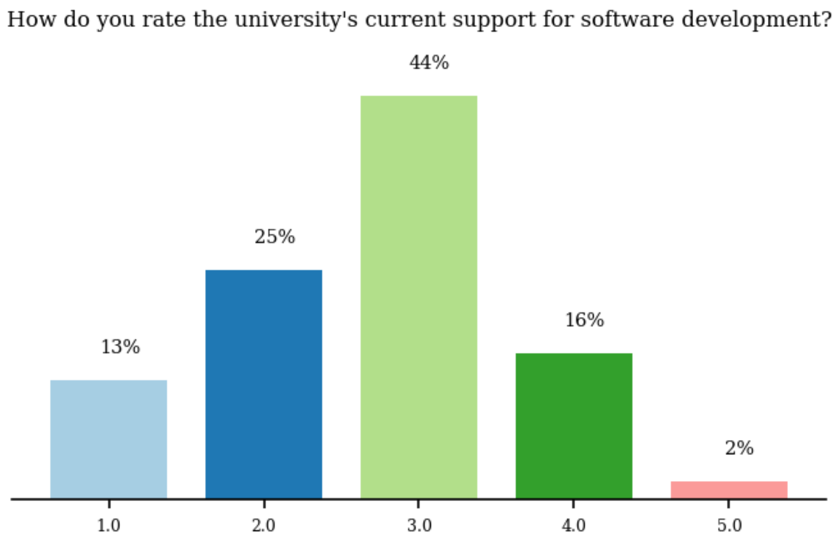
11.4 Appendix D

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



11.5 Appendix E

We investigated how researchers who develop software felt about the University's current level of support for software development:


How do you rate the university's current support for software development?