Science & Technology Facilities Council

EPSRC
Engineering and Physical Sciences Research Council

Analysis of the 2016 SESC Survey on Continuous Integration practice and views in the UK

Catherine Jones, Steven Lamerton, Gemma Poulter & Alan Kyffin

May 2017

Science & Technology Facilities Council

EPSRC Engineering and Physical Sciences Research Council

Document History

Version	Date	Author	Notes
V0.1	1/12/2016	Catherine Jones	Initial draft
V1.0	4/12/2016	Catherine Jones	Checked & edited for the Advisory Board
V2.0	29/3/2017	Catherine Jones	Extended with additional analysis and feedback from SESC team.
V3.0	16/5/2017	Catherine Jones	Revised figures and final conclusions

Science & Technology Facilities Council

EPSRC Engineering and Physical Sciences Research Council

Contents

Science & Technology Facilities Council

EPSRC
Engineering and Physical Sciences Research Council

1. Introduction

The Software Engineering Support Centre ran a survey on Continuous Integration tool usage and gathered views on the developments planned for the SESC Build Service. It sought to understand the current state & usage of CI/automated testing in the academic research software community and to gain some feedback about the potential usage of services being developed by SESC for that community. This document analyses the results.

The survey ran during November 2016. A total of 83 responses were received. This was within the success criteria range for the survey. The invitation to participate was sent to

- the current Collaborative Computational Projects via EPSRC and the CCP Chairs;
- the Research Software Engineering Community using the RSE mailing list and the RSE general Slack channel.

As the survey was anonymous it was not possible to differentiate by which route that the respondents had heard of the survey, and there will be overlap between the two communities.

The Software Engineering Support Centre is funded by EPSRC and is run by the Software Engineering Group in the Scientific Computing Department at STFC.

2. Information on respondents

The survey asked about the role and funder of the software being developed to be able to identify the perspectives of the respondents. We also wanted to establish the interest in the service beyond the current EPSRC funding remit.

Figure 1 *Roles of respondents* shows the roles that the respondents reported. The question allowed the respondent to choose more than one answer. Just under half identified with RSE/Software Developer. As this survey was aimed at those who actively develop software, the responses may be considered to be fairly representative of both those who do create software and those who may be responsible for technical decisions.

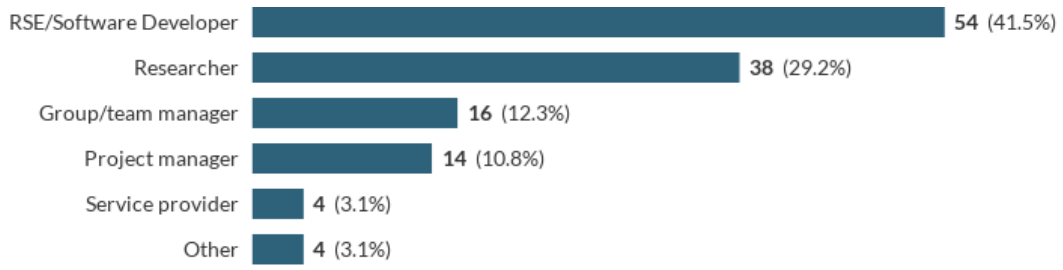

Figure 1 Roles of respondents

The four roles represented in "Other" were a PhD student, System Admin (2) and a University academic with an interest in software development.

2.1 What is the subject domain of the respondents?

The subject domain of the respondents was an optional question and not all respondents completed this. The results have been categorised into high level domains. Although physical sciences dominate the reported subjects, nearly 20% reported working across many subject domains.

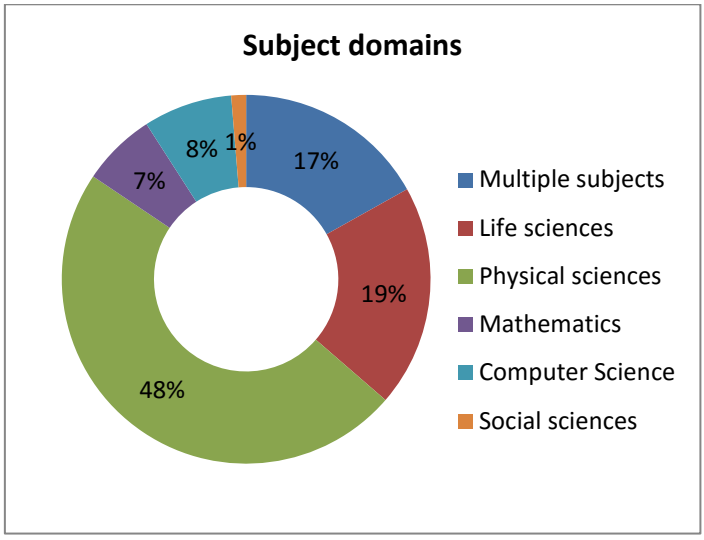

Figure 2 Subject domains

2.2 Who funds the software?

The survey asked about who funded the software to understand the wider landscape and the overlap between respondents and the SESC EPSRC supported remit.

The Research Councils fund 44.6% of the software being developed by the respondents, which is to be expected from the community surveyed. A significant proportion of respondents are working on multiple software projects funded by a range of sources.

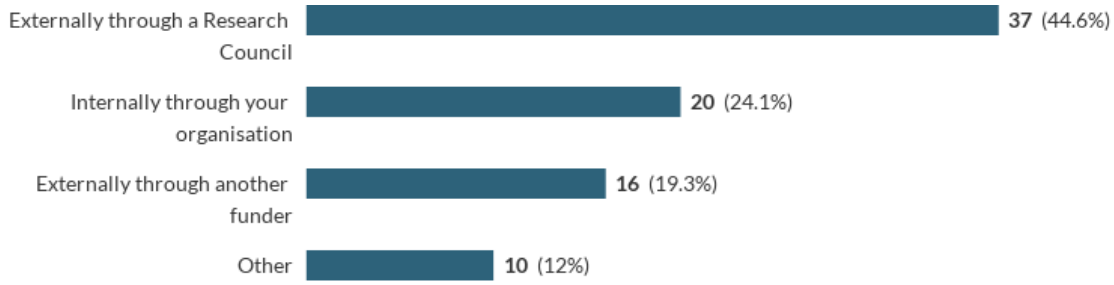

Figure 3 Funders of the software being developed by the respondents

Other external funding sources included the Wellcome Trust, European Research Council, the EU, National Institute for Health Research and Jisc.

A further clarification question was asked to identify which specific Research Council was funding work. All seven research councils were reported; but as to be expected, the largest funder of software reported in the survey was EPSRC. We were pleased to see all Research Councils represented as this supports the 2014 SSI survey[1] on the use of research software findings that researchers use and create research software across the breadth of subject domains.

[1] https://www.software.ac.uk/blog/2014-12-04-its-impossible-conduct-research-without-software-say-7-out-10-uk-researchers

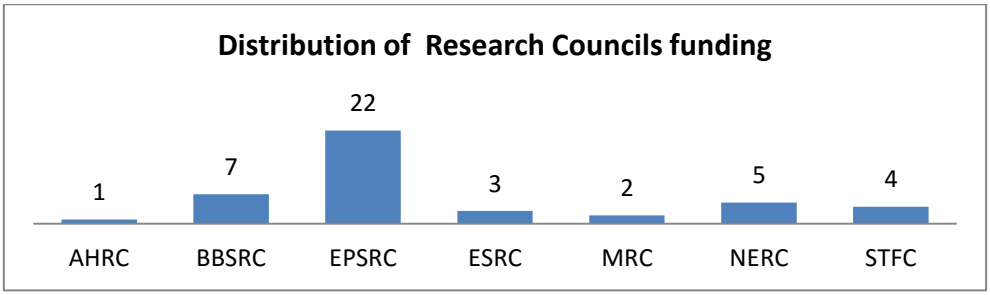

Figure 4 Research Councils funding the software being developed

Note that some respondents named more than one Research Council as funding their activities.

Although the survey size is relatively small, it is clear that the respondents are the intended audience for the survey.

3. Current Usage

The second section of the survey was interested in the current automated testing/CI practice and HPC usage in the community. This would enable SESC to test our hypothesis that there is need for a developed SESC Build Service which provides testing facilities on HPC resource provided centrally for the UK academic community.

3.1 Programming Languages in use

The respondents were asked about the programming languages they used regularly and could choose more than one answer. As in the SESC/CCPForge user survey (2015), Python, followed by Fortran, C and C++ were the top four languages in use.

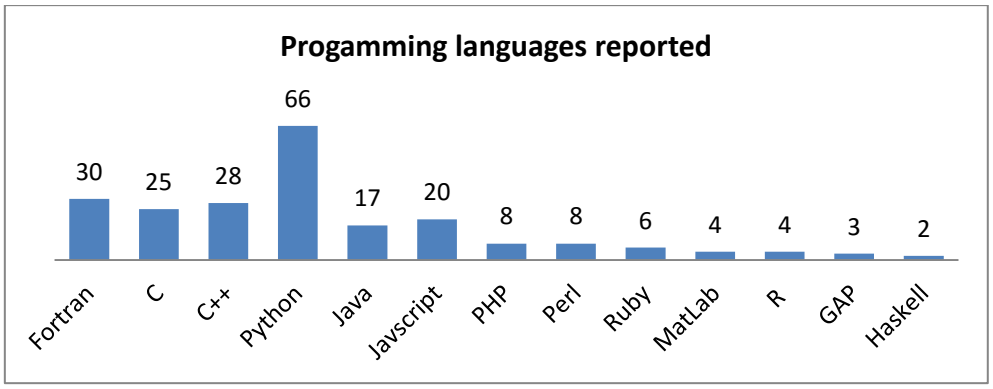

Figure 5 Programming languages in use

In addition to the languages in Figure 5 *Programming languages in use* there were single mentions of Bash, C#, Clojure, Lua/LuaJIT, Mathematica, Julia, Octave, OpenCL, OpenMP, CUDA, Scheme, Idris and SQL.

As this survey addressed a wider range of software developers than the CCP community using CCPForge who were the focus of the 2015 SESC survey, there was a wider range of other languages reported.

To see how representative this is of the wider academic research community, we compared the results with the SSI's *Software used in Research based on combined surveys*[2] This study looked at over 2958 responses from 1261 survey participants, so a much bigger sample than this survey. The SSI collected the data on software used in research through asking "what software do you use in your research" and the results of five surveys, run between 2014 and 2016 were combined.

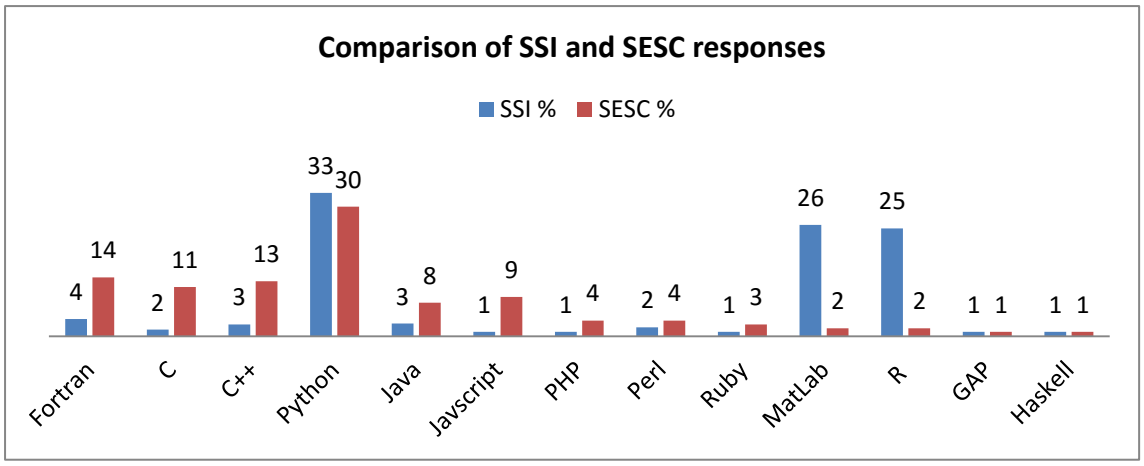

Figure 6 Programming languages reported in SSI and SESC work has calculated the percentages for each language against the total of the languages for both pieces of work, using languages reported more than once in the SESC survey as a starting point. This enables comparisons to be made and adjusts for the difference in size and scope of the SSI work against the much smaller SESC survey.

In this analysis, Python is the most popular language reported in both and is around a third in both datasets. All the languages in the SESC survey apart from Julia and OpenCL were reported in the SSI data.

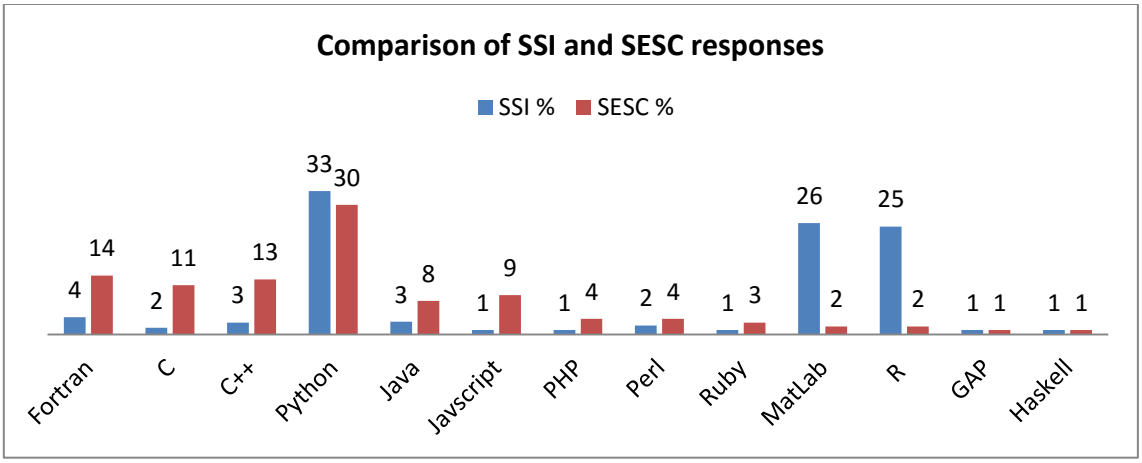

Figure 6 Programming languages reported in SSI and SESC work

The main difference is the divergence between the popularity of MatLab and R in the SSI data and Fortran, C and C++ in the SESC data. It is likely this is due to the wider audience that the SSI data was drawn from compared to the more focussed, computational community that the SESC data was

[2] Hettrick, S., Philippe, O., Chue Hong, N., Sufi, S., Silva, R., & Peru, G. (2016). Software used in research based on combined surveys [Data set]. Zenodo. http://doi.org/10.5281/zenodo.60276

drawn from. However for languages such as Java, there was a greater representation of this in the SESC data and some of the more specialised languages there was a similar level of reported use.

3.2 Continuous Integration and Build services

65 respondent (75%) use CI services in their software development process, which means that a quarter of the respondents don't use a CI service. If the survey results are filtered by job role, then 44 (81.5%) of those identifying themselves as Software Developers/RSE do use CI services.

As the survey didn't ask for the reason for usage of these services, it is not possible to ascertain what factors influence the decision. Potential factors, from our experience, may be the size of the development team (the more there are the more useful CI can be perceived to be); cultural expectations (the greater the number of formally trained developers is likely to led to a more formal approach to testing) and access to services relevant for the technology in use.

The survey asked who provided the services and they fall into the following categories:

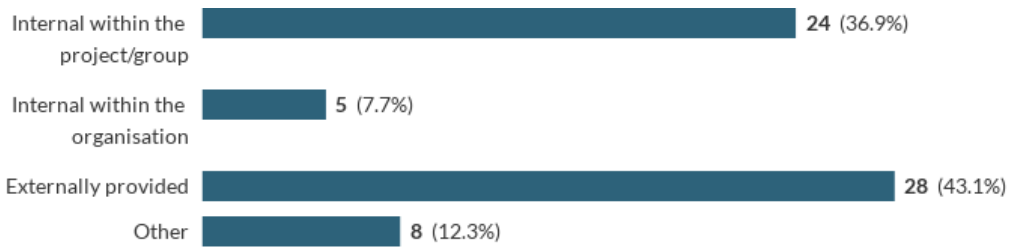

Figure 7 Provision of CI services

Those who chose the "*other*" option used a combination of internal AND external tools. On reflection it is obvious that developers who work on more than one project might use a combination of CI tools and the survey should have reflected this. There were a wide range of CI tools reported, but the two most popular tools by far were Travis and Jenkins.

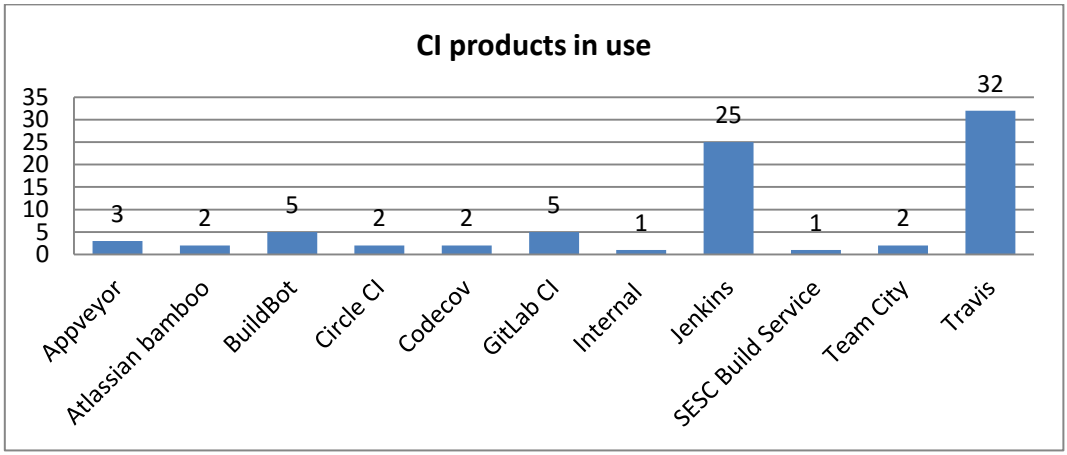

Figure 8 CI services in use

Figure 9 *CI service provision* below shows that there is a fairly even split between those who use internal services and those who use a free external service, which reflects the use of the free Travis service.

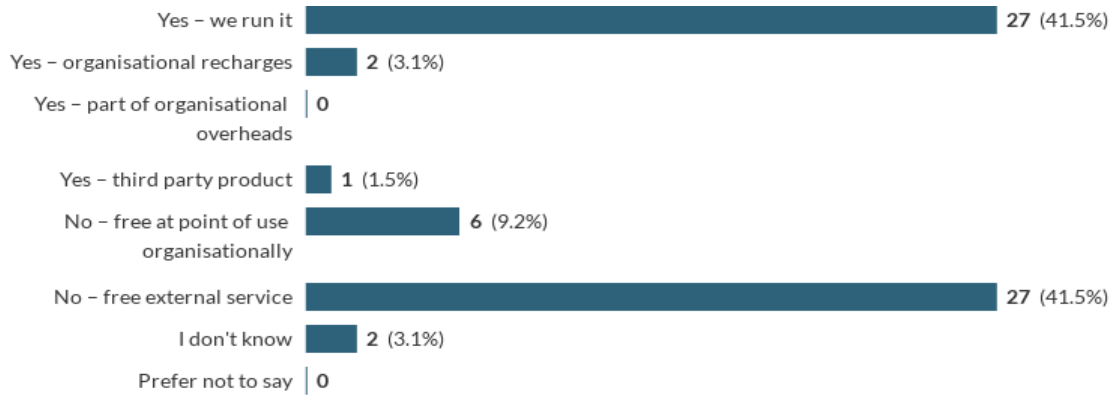

Figure 9 CI service provision

Figure 10 *External/Internal split* below, shows for the four most popular services/products the split between products such as Jenkins which tend to be run in-house and the externally provided Travis service. Of the other products mentioned in Figure 8 *CI services in use* there is a fairly even split between external products and internal products.

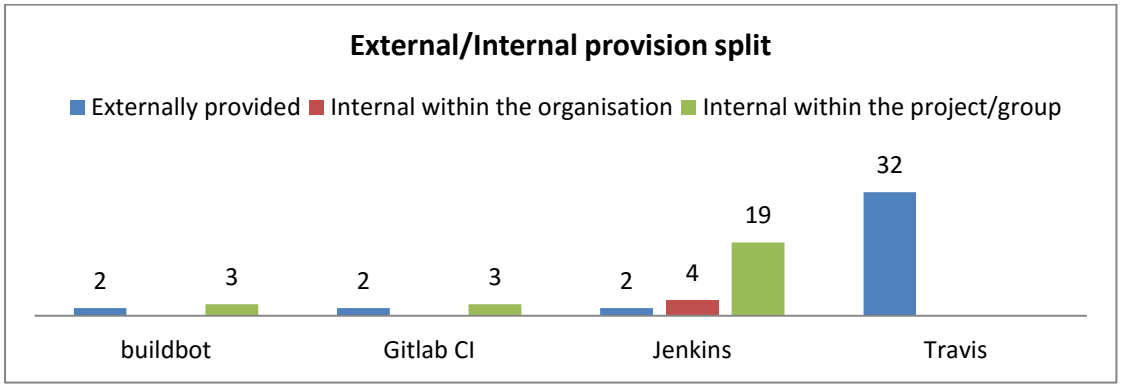

Figure 10 External/Internal split

3.3 Tools needed to use CI services

The survey asked for the three most important tools needed to build and test the respondents' software. Here we were looking to see what additional tool support a service being provided to others would need to provide. There were a large number of responses and we have grouped the tools into categories to aid understanding. As might be expected access to compilers, specific languages and specific libraries are the top three areas.

Figure 11 *Trends of tools by importance* below shows the relative importance of the categories by reported importance. This demonstrates that compilers are the first important tool for a large number of respondents, and are the most important tool overall, whereas specific libraries and testing tools are important but are not the first tools mentioned. This data gives an indication of relative importance of the various tools and shows what our respondents thought were most important.

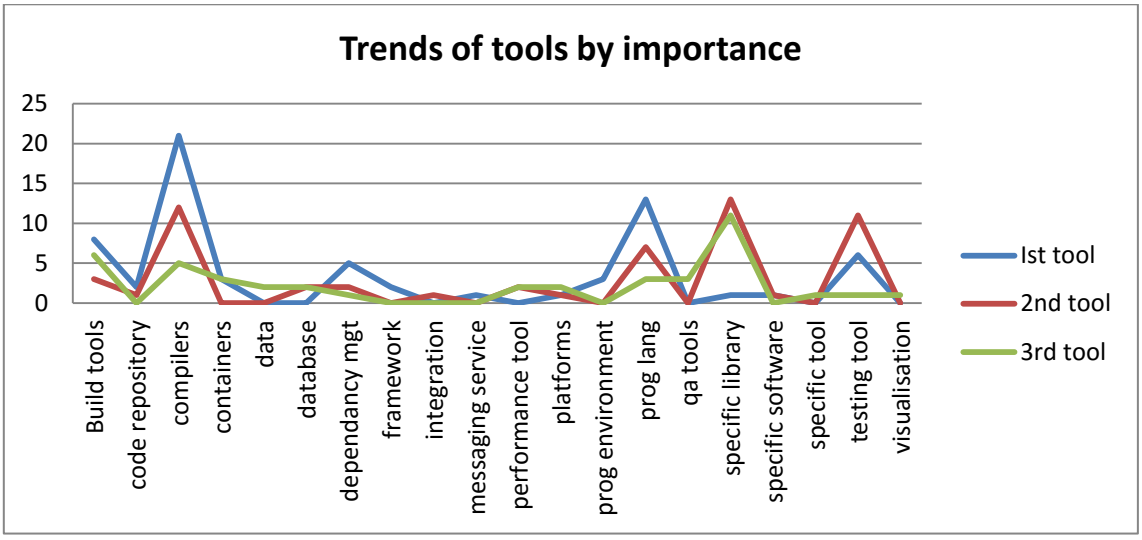

Figure 11 Trends of tools by importance

3.3.1 Compilers

Compilers form the bulk of the named tools over the three tool questions. Figure 12 *Split between compilers for* different languages shows that there is a fairly even split between C/C++ compilers and Fortran compilers.

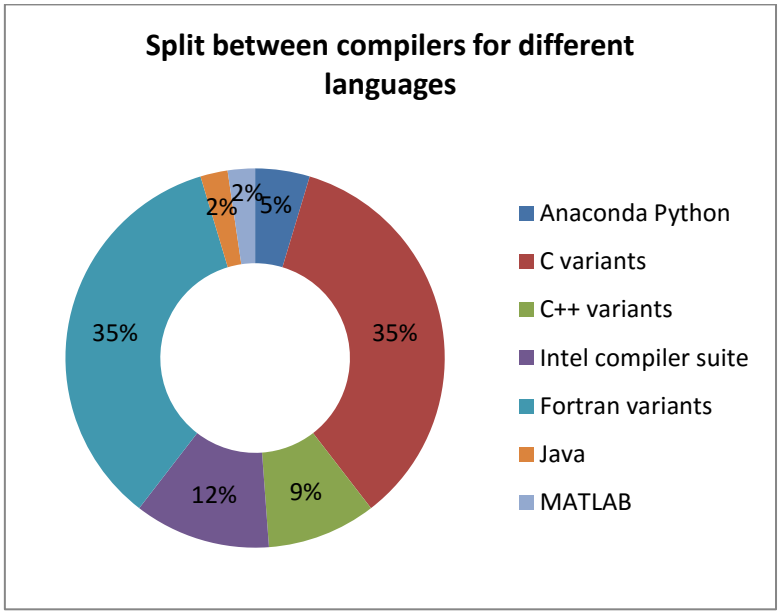

Figure 12 Split between compilers for different languages

Looking at C & C++ and Fortran compilers in more detail Figure 13 *Specific C/C++ and Fortran compilers mentioned* details the breakdown within the language for named compilers.

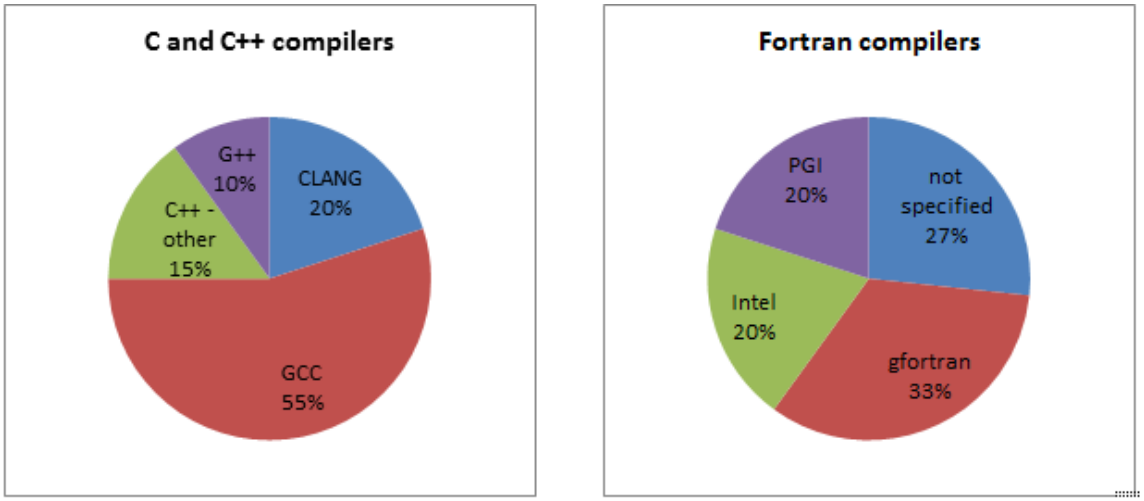

Figure 13 Specific C/C++ and Fortran compilers mentioned

This data will enable SESC to put together a plan for ensuring the main compilers reported are available and the order in which they were identified can also help with prioritisation. We are also looking in more detail at specific libraries and testing tools.

3.4 Operating Systems needed for testing

SESC was interested in the types of Operating System needed for effective testing of the software being developed. Figure 14 *Operating systems needed for testing* below shows the distribution of operating systems used (more than one option could be chosen). This shows a strong preference for Linux, but also a good showing for Windows and Mac OS.

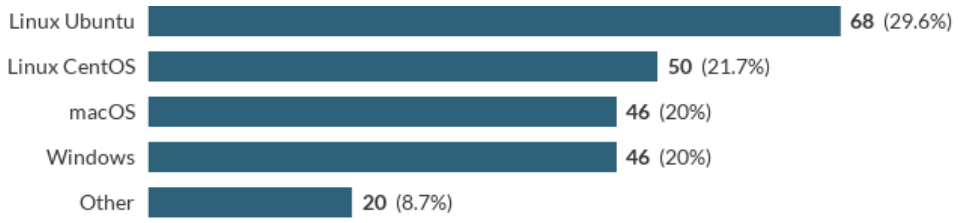

Figure 14 Operating systems needed for testing

Of the other OS, 15 were variants of Linux (Scientific Linux or RedHat), two were BSD and the others were Android, Arduino and Cray offering.

The survey then asked what was the most important operating system needed for testing, here Linux variants dominated.

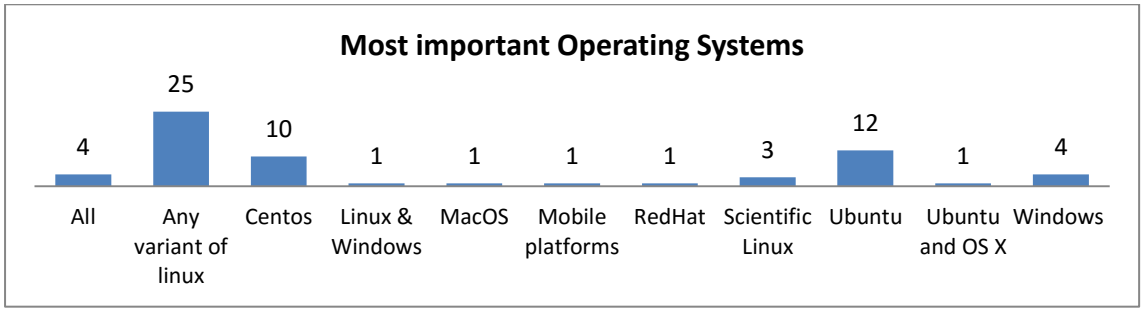

Figure 15 Most important Operating Systems

3.5 HPC Usage

As the future development plans for the SESC Build Service are focused on enabling testing on HPC resources, the survey wanted to get a feel for the number using HPC resources; the table below shows that 75% of the respondents did use HPC in one form or another. The other respondents used overseas (Titan) or more than one of the options. This is a very high percentage of HPC users, this may be because of who the survey was sent to, or that the description of the survey encouraged HPC users to complete it or perhaps HPC users have a more formal approach to automated testing. It is not possible to know why those completed it did so.

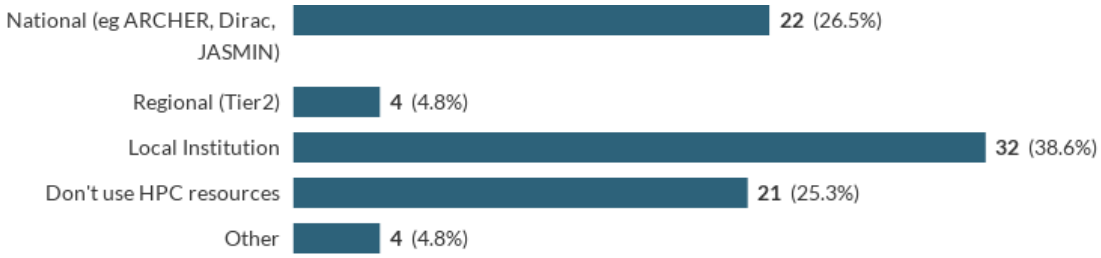

Figure 16 Use of HPC resources

Having established that some of the respondents used HPC resources, we wanted to ascertain how many of those would be interested in using a central service to test their software. As can be seen from Figure 17 *Level of reported interest in HPC testing* below there was a large amount of interest shown from the respondents. This is promising for the SESC Build Service plans.

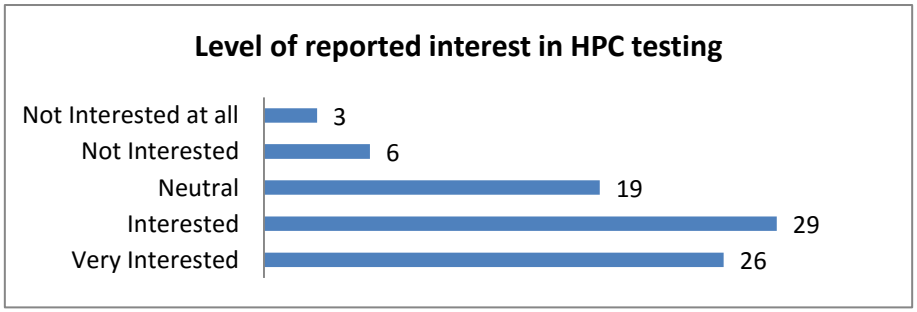

Figure 17 Level of reported interest in HPC testing

3.5.1 Novel architectures or testing

Another of the key SESC aims is to explore enabling testing on novel architectures, so the survey asked about the novel architectures that people might want to use and whether they would use a central service for this.

The categories of novel architecture reported fall into the categories below.

- ARM (includes AArch64)
- Xeon Phi (includes Knights Landing, KNL, Knights Corner, MIC)
- GPGPU (includes NVIDIA, AMD)
- FPGA
- Power (includes Power8, Power8+)

These are well represented in the newly announced National Tier 2 infrastructures[3]. ARM: Isambard; Xeon Phi: Peta-5 and Isambard; GPGPU: Jade, Peta-5 and Isambard and Power8: HPC Midlands Plus.

The respondents of the survey were interested in testing software against these architectures but fewer than were interested in HPC as a whole.

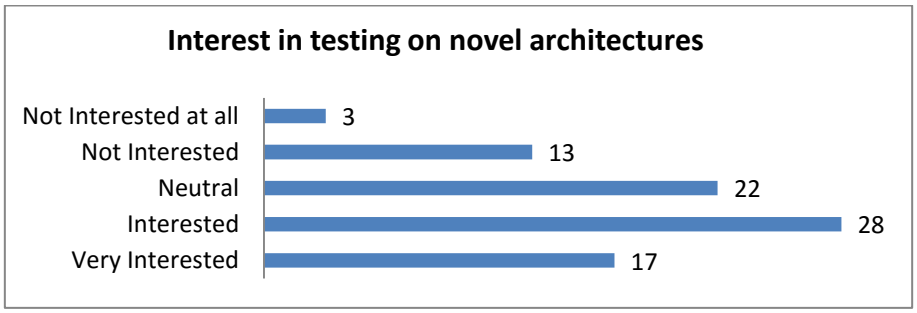

Figure 18 Interest in testing on novel architectures

4. Views on SESC development plans

The final section was designed to get views on the SESC planned developments to sense check them and establish whether there would be a viable service. The views on HPC and novel architecture testing have been covered in section 3.5 HPC Usage.

4.1 Authentication systems

The first question was on the authentication systems. This was asked to see if the SESC preferred starting point of Shibboleth would be used by the community. There had been some feedback that Shibboleth would restrict the potential audience from the SESC Informal Technical Network who provide SESC with feedback. As can be seen from Figure 19 *Views on authentication systems* below, Shibboleth and OpenID are very evenly matched. As expected, CCPForge credentials were the least preferred option, which supports our technical decision to decouple the two services. A surprising number were happy to use a service specific user name. As a result of this survey we are continuing with Shibboleth development but with a view to rolling out OpenID afterwards.

[3] https://www.archer.ac.uk/community/champions/workshops/leeds_feb2017/talks/L02_Tier2_Champions.pdf

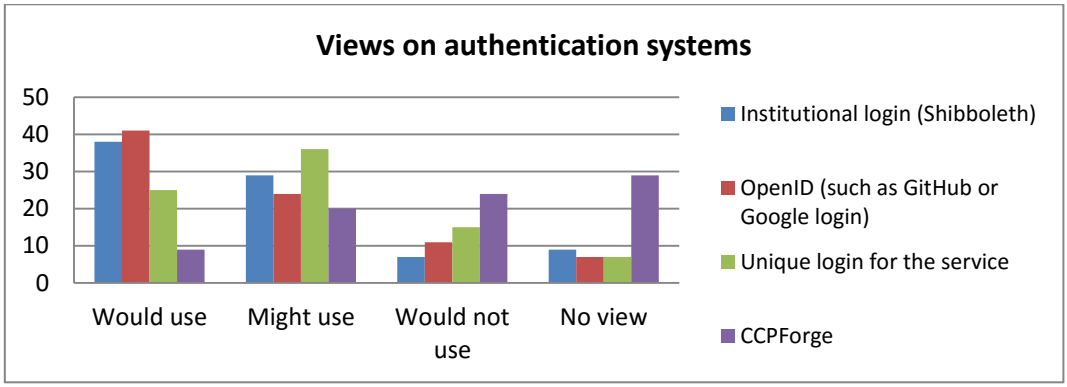

Figure 19 Views on authentication systems

The data was captured using a question where each authentication was given a rating, but the ratings for each option were independent of each other, further analysis of the responses to this question have identified the following trends/relationships.

Of the relative relationship between Shibboleth and OpenID:

- Twenty-three respondents who would use either Shibboleth or OpenID.
- Four respondents who would use OpenID but not Shibboleth
- Five respondents who would use Shibboleth but not OpenID
- Of those Software Developers/RSEs who responded, twenty-six would use Shibboleth compared to twenty-five who would use OpenID.

When analysing the data, the following additional facts/patterns were established:

- Four respondents preferred the alternatives and would not use Shibboleth or OpenID. This was a surprise.
- There were five respondents who had "no view" of all the options. Taking this into account when looking at the responses, it means that a large portion of respondents had no view on CCPForge, which is to be expected as it is a niche service.
- There were two respondents who would not use any of the authentication systems, thus leading to the conclusion that they would not use a central service.
- There were nine respondents who would or might use all the options, thus not showing a preference for any particular authentication system.

The overall analysis confirms that Shibboleth and OpenID are the main two contenders for authentication, and as there is not a great difference between the results, and Shibboleth has an advantage of being linked to UK academic institutions, then the result is that SESC will continue to prioritise Shibboleth over OpenID.

4.2 Service Characteristics

The next series of questions were to identify what features and characteristics of a centrally provided service would be most important. The questions addressed service provision, technical requirements and expertise provision.

As can be seen from the figures below the four most important areas are ease of use, longevity of service, well known framework and range of operating systems. The first three relate to the service

and are understandable, to use an automated testing system the developer must invest their time and they need to be reassured that the service will be there for some time, that product used is not a niche one and that no more time than necessary will be invested to get the service running.

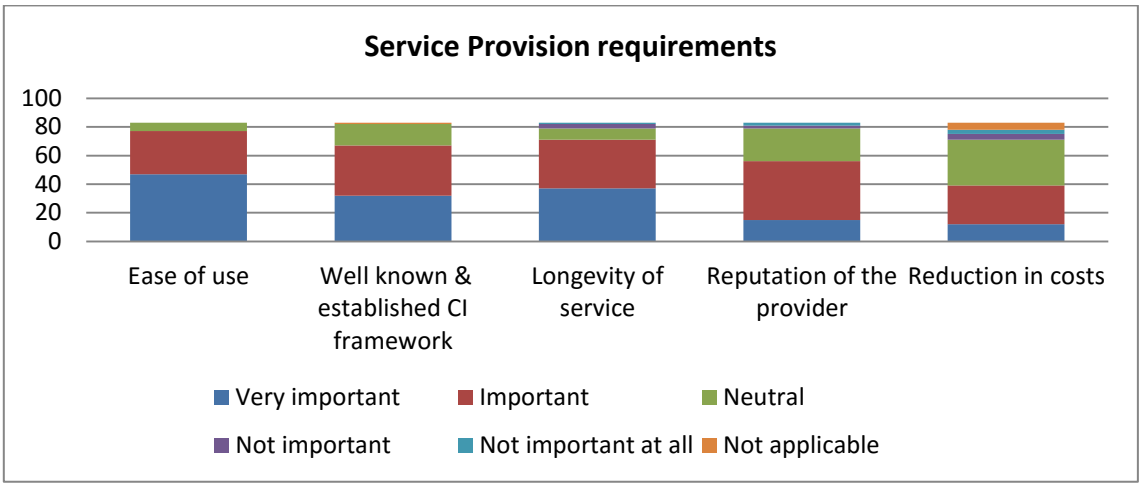

Figure 20 Service provision requirements

As Figure 20 *Service provision requirements* demonstrates any central service must be easy to use and well established within the community. There is some interest in a reduction in costs, but it is not the main driver in deciding whether to use an external central service.

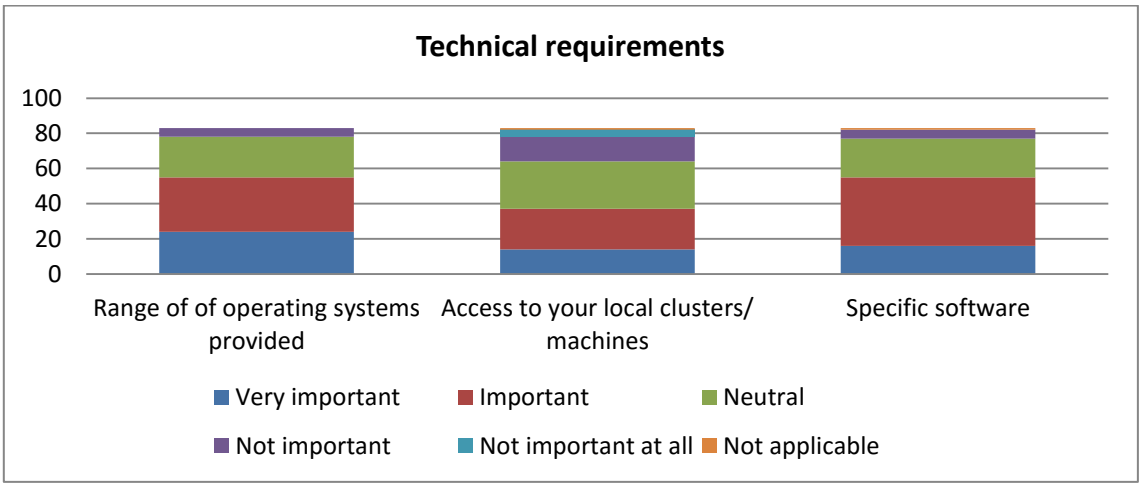

Figure 21 Technical requirements

As Figure 21 *Technical requirements* demonstrates a central service would need to support the technical requirements which have been explored elsewhere in this report, but access to local machines is not as important.

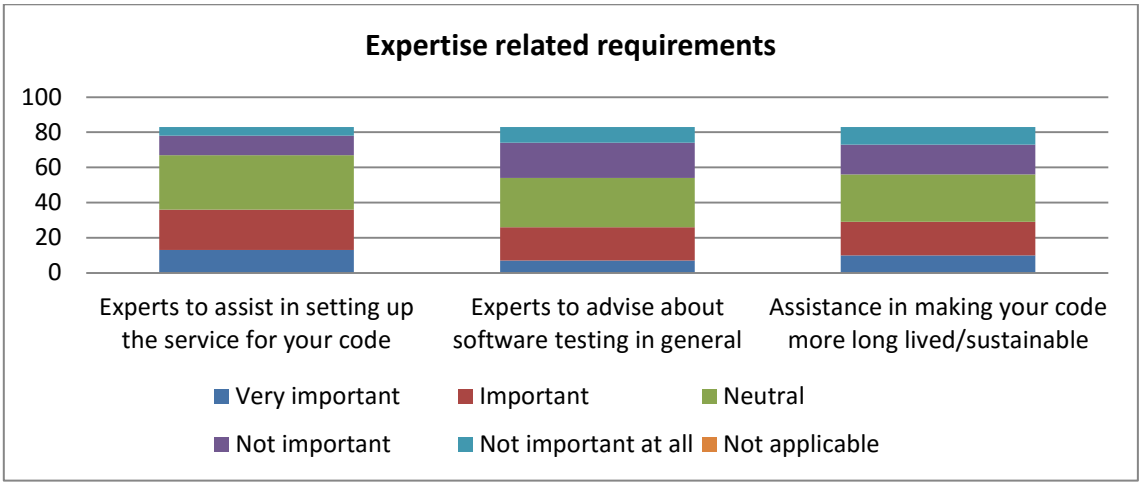

Figure 22 Expertise related requirements

Finally Figure 22 *Expertise related requirements* shows lesser importance rating than other areas previously discussed, although there was still interest in these areas. It may be that the expertise areas are perceived to have an overlap with other programmes such as eCSE or SSI Open Calls, or that as previously discussed this survey had a high proportion of HPC users and so are more experienced and less likely to need expert assistance.

These results will feed back into the planning for SESC developments, it establishes that there would be some take-up for these more consultative services, but they are not as important as the technical requirements.

5. Free text comments

The survey offered the respondents an opportunity to provide free text comments to assist in the interpretation of the results. There were a relatively small number of free text comments at the end of the survey. They can be categorised into the following areas:

- It is difficult to be better than the free tools (3)
- Don't limit to CCPForge/CCPs (2)
- Intel compiler on CI services is a limiting factor
- Good idea for HPC integration(2)
- Being able to deploy through packages; containerisation; dependency management
- Ability to learn new complicated systems when running a range of small software developments

We are well aware of the issues of providing a service which is not as good as existing free services. It is also in our plans to decouple the service from CCPForge. It was encouraging to have some positive comments on the direction of our plans.

6. Conclusions

The survey attracted 83 responses from the CCP and RSE communities, this was within the success criteria. Whilst this is still a relatively low number compared to the number of software developers/RSEs/Computational scientists in the UK academic community, it is a good sample to be

able to draw some conclusions. The response rate demonstrates that there is enthusiasm within the community to provide their views on this subject.

The survey aimed to establish the current practice of automated testing and to ascertain if the SESC development plans would resonate with the community. Although the analysis has identified areas where further questions could have been asked, there is a wealth of information on the practices and some useful feedback for the SESC team.

Key headline results are that:

- 75% of all respondents are already using CI systems
- 81% of those who identified themselves as software developers/RSEs are already using CI systems
- A significant minority of respondents use more than one CI system
- All of the seven UK Research Councils are represented, with EPSRC being the biggest funder
- Python, Fortran, C and C++ are the most common languages reported; this is not the same as SSI findings.
- There are a high proportion of HPC users in the sample
- Travis and Jenkins are the most common CI frameworks in use
- Compilers and specific libraries are the most important tools needed for automated testing, with an equal split between Fortran and C & C++.
- There is an interest in HPC testing, and a smaller interest in novel architectures
- Shibboleth and OpenID are of equal importance as authentication mechanisms

Based on the survey results, for the SESC Build Service to be a success then it needs to:

- be seen to be sustainable and available over a long time period
- provide testing access to national and regional HPC and build good relationships with these
- provide the right support for programming languages through access to compliers and libraries
- provide access through Shibboleth and OpenID
- be easy to use, through the CI system chosen and the support provided
- explore additional services based on automated testing expertise, linked to other initiatives in this area

The survey will inform SESC's future planning. The SESC team would like to thank all those who took part for their time and EPSRC and the SSI for their advice regarding the survey design.

7. Appendix I Additional analysis of the tools data

The bar charts below show the underpinning data for Figure 11 *Trends of tools by importance*.

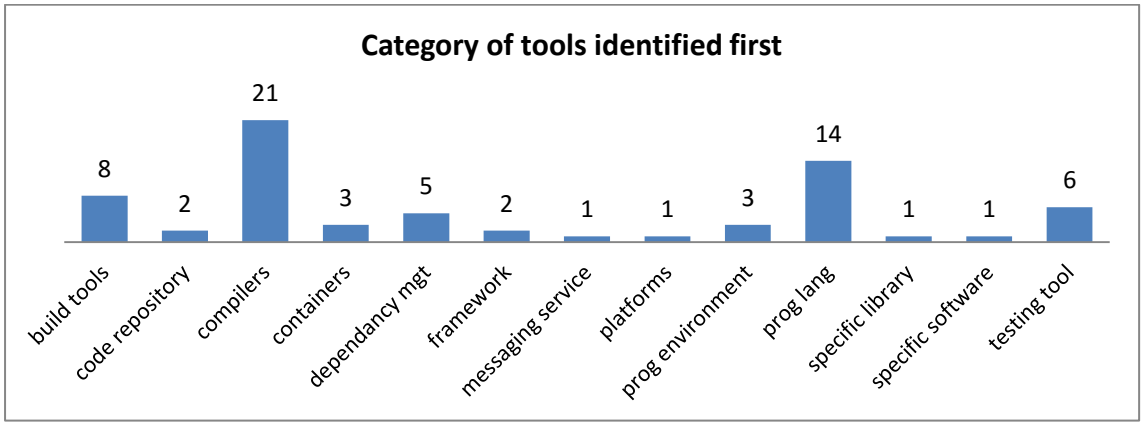

Figure 23 First Tools used/needed in automated testing

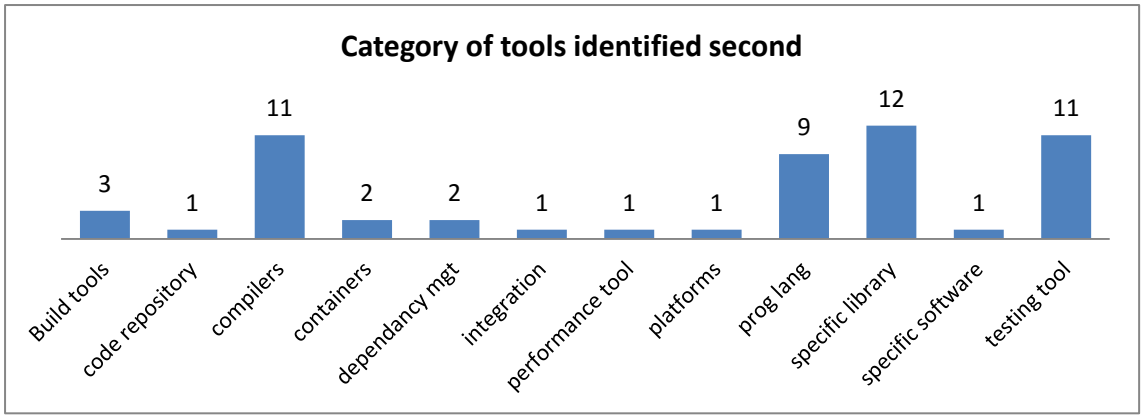

Figure 24 the Second tool used/needed in automated testing

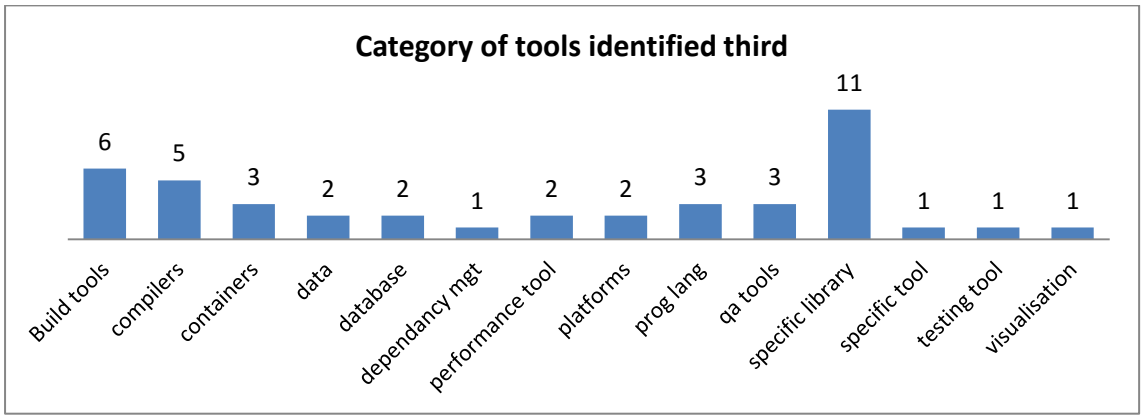

Figure 25 the Third tool in use/needed for automated testing