How to encourage scientists to publish their code

A dissertation submitted to the University of Manchester for the degree of Master of Science in the Faculty of Engineering and Physical Sciences

2015

By

Rawan A. Sanyour

School of Computer Science

Table of content

Words count 27,326

List of Figures

List of tables

Abstract

The recent rapid growth of computational science, including computer science, chemistry, bioinformatics, mathematics, physics, and human science, has raised a number of new challenges about reproducibility. In spite of the fact that science essentially relies on computer code that is used to set up and run the data analysis, this code is often not included in publications. However, there is an urgent need to include these details since withholding them can negatively influence the credibility of scientific works, prevent efforts to verify and validate the work, and impede the process of building on that work.

The dominant culture in the scientific community views that it is not necessary or expected to release accurate code because the experiment's idea and results are perfectly clear. To change the resistance that scientists show regarding the issue of code sharing, efforts must be made at several levels, including: journals, research funding agencies, and the scientist themselves who must work hard to change their publishing attitude and start to practice openness in their publications. This project has produced technical recommendations, designed through interviews with scientists and research software engineers that aim to address these technical barriers. The recommendations have been evaluated through a large scale survey with computational scientists that shows a significant level of support for these recommendations which indicates that scientists are struggling and need help to overcome the technical issues that they face during code publishing process. The value of the results is demonstrated by the interest of the Software Sustainability Institute who is publishing them on their website.

Declaration

No portion of the work referred to in the dissertation has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.

Intellectual Property Statement

I. The author of this dissertation (including any appendices and/or schedules to this dissertation) owns certain copyright or related rights in it (the "Copyright") and s/he has given The University of Manchester certain rights to use such Copyright, including for administrative purposes.

ii. Copies of this dissertation, either in full or in extracts and whether in hard or electronic copy, may be made only in accordance with the Copyright, Designs and Patents Act 1988 (as amended) and regulations issued under it or, where appropriate, in accordance with licensing agreements which the University has entered into. This page must form part of any such copies made.

iii. The ownership of certain Copyright, patents, designs, trademarks and other intellectual property (the "Intellectual Property") and any reproductions of copyright works in the dissertation, for example graphs and tables ("Reproductions"), which may be described in this dissertation, may not be owned by the author and may be owned by third parties. Such Intellectual Property and Reproductions cannot and must not be made available for use without the prior written permission of the owner(s) of the relevant Intellectual Property and/or Reproductions.

iv. Further information on the conditions under which disclosure, publication and commercialization of this dissertation, the Copyright and any Intellectual Property and/or Reproductions described in it may take place is available in the University IP Policy (see http://documents.manchester.ac.uk/display.aspx?DocID=487), in any relevant Dissertation restriction declarations deposited in the University Library, The University Library's regulations (see http://www.library.manchester.ac.uk/about/regulations/) and in The University's Guidance for the Presentation of Dissertations.

Acknowledgments

I would like to express my sincere gratitude and appreciation to my supervisors Dr. Caroline jay and Robert Haines for their endless support and valuable advice which has helped me completing this project.

I would like also to extend my gratitude to my dear husband, Khalid Aljohani, for his unconditional support and patience during this period, without his help, I could not go through all this.

Finally, I gratefully wanted to thank my dear family- my mother, father, brothers and sons for their endless love, support and encouragement.

Chapter 1

1.1- Introduction

A scientist's inability to reproduce the results of some research studies is a significant barrier to progress in science. The word "reproduce" means the results obtained through a specific experiment can be recreated easily by other individuals in addition to the author. Recently, "black box" science which has hidden details has been considered the biggest obstacle to reproducibility because without the code, all efforts to reproduce and gain benefits from computational experiments will fail. Over the years, it had been shown that each generation of researchers builds their work upon previous scientists' achievements. Hence, the need for transparency in scientific fields is urgent. The numerous benefits, which could be gained by releasing the scientific experiments' details such as analysis code to the scientific community and the author himself as well, make such details availability an urgent demand (LeVeque (2012) and Donoho (2010)).

1.2- Aim and objectives

1.2.1- Aim

The aim of this research was to provide valuable recommendations to scientists to encourage and motivate them to publish analysis code publically alongside their academic papers thus improving the reproducibility process and benefitting the scientific field. This aim has been accomplished by conducting an investigation to understand both the technical and cultural reasons behind the scientists' decisions regarding whether to publish their analysis code alongside their results and developing a set of recommendations to address the identified barriers and evaluating these recommendations with scientists.

1.2.2- Objectives

- To perform a systematic literature review based on the research question to investigate why scientists do not publish their code, to determine what barriers they face if they intend to do so.
- To uncover the factors and cultural and technical issues affecting scientists' decisions regarding publishing their experimental code. This has been achieved by conducting an interview study with several scientists who are publishing their code.
- To examine and address these technical issues in more detail. This has been accomplished by conducting another interview study with the Research Software Engineering group.
- To develop a series of recommendations that will be evaluated by a group of scientists from different computational fields to determine how effective they are perceived to be. This has been done by conducting a questionnaire survey
- To present these recommendations to the experts of the Software Sustainability Institute (SSI), which is an institution that employs a great team of experts with extensive backgrounds in several disciplines, such as software development, research facilitation and community engagement, to support researchers to improve and increase the quality of their software, as well as to train people to have skills that enable them to develop reproducible work. This has been achieved by publishing an invited post in the SSI blog.

1.3- Dissertation outline

The structure of the thesis will be as follows:

- **Chapter 2. Background:** In this chapter, the scientific literature was surveyed to situate the investigation in a wider context regarding the related work. A systematic literature review was conducted to gain a deeper insight into code publishing and all of the related issues that could influence such a decision.

- **Chapter 3. Method:** This chapter briefly introduces the methods that have been used to collect and analyse the data to answer the research question.
- **Chapter 4. Investigation of the barriers encountered by scientists trying to publish code:** Within this chapter, the data, which has been collected through the interviews with scientists who already publish code, were thematically analysed to explore issues in relation to code publishing from their viewpoints.
- **Chapter 5.Investigation into research IT professionals' views on publishing code:** In this chapter, the data collected from these interviews with the Research Software Engineering group to address the technical issues in detail were also thematically analysed to develop evidence-based recommendations to help scientists in the process of publishing code.
- **Chapter 6. Questionnaire analysis**: In this chapter, the data that have been collected through the survey have been analysed to determine whether they were useful and can facilitate the publishing process.
- **Chapter 7. Recommendations:** in this chapter, a set of recommendations have been proposed to help scientists overcome the identified technical barriers and, thus, help them to publish their experiments' code.
- **Chapter 8. Conclusion:** A brief conclusion, project limitations and future work have been summarised in this chapter.

Chapter 2

2- Systematic literature review

2.1- Introduction

For decades, scientists in the academic community published only a description or a pseudo code of their algorithms and experiments instead of releasing the source code. Over the passage of time, problems with these alternatives have begun to rise. Thimbleby (2003) has illustrated an example which explains the problem of pseudo code and why it is not suitable to be presented instead of source code in scientific papers. In the case of Porter's stemming algorithm which was invented to find the canonical form of any word, for example programmable and programming words will both refers to "program" word. The problem of Porter's algorithm is that, it was published in a nonprogrammable form thus many incorrect implementations were produced based on that form. Porter himself recognized misunderstanding as one of his algorithm's problems. Consequently, many researchers have presented incorrect results by applying an incorrect version of Porter's stemming algorithm in their published work. Regarding this, Thimbleby (2003) believes that it is much easier to provide the source code on a website, for example, rather than trying to extract a description or pseudo code from the source code.

Freely providing code in academic papers is a controversial point of debate in the scientific community and the question of whether or not scientists should release their research code alongside their academic published papers has not yet received an adequate answer.

In computational science fields, which now span all fields of natural and human science, as well as data science, the vast majority of scientists resist the idea of releasing their working code publicly; they rarely provide their working code with their scientific papers (the reasons for this will be covered in the literature review).

However, today, things are starting to change. The scientific community is calling for more transparency in publications and is pressuring scientists and researchers to change their

attitudes regarding code sharing. For example, in an article published in ***NATURE***, Nick Barnes (2010) started his argument about the importance of releasing working code by saying that:

"Freely provided working code, whatever its quality, improves programming and enables others to engage with your research."

As a professional software engineer, Barnes (2010) has asserted that practicing openness can improve the code quality, and give others the capability to validate the work and participate in it, thus leading to significant improvement in the software industry.

Barnes has presented his own experience as a volunteer to improve the code published by NASA that is used to report global temperature. The code was initially messy and full of bugs. After Barnes and others had rewritten the code, it became easier for non-experts to understand and run Barnes (2010).

In this investigation, attempts were made to generate a set of valuable recommendations to be used as part of a long term solution that could motivate scientists to share their code publicly.

2.2- Research question

The aim of this review was to address the question of why scientists do not publish their code. It also covers the theory of reproducibility, which is a new discipline that has recently come into the view of computational science.

The review is based on various publications that have carried out multiple studies and investigations on the reasons behind the scientists' decisions not to publish their experiment's code, the barriers that prevent them from publishing, the urgent need for code publishing, and the benefits gained from such publishing. It followed a systematic review protocol as described in the following sections.

2.3- Search strategy

2.3.1- Data collection

2.3.1.1- Databases

Several electronic databases have been reviewed to extract studies that are relevant to the research questions, these databases are: Google Scholar, ACM (Association for Computing Machinery) library, Web of Science, Science Direct, IEEE Xplore, Scopus, Springer Link, Ethos British Library, Open Access Library, refseek, and dblp (the computer science bibliography). The reference lists of the found articles were also searched to try to find additional articles that are related to the research area. It has been noticed that some of the included articles are cited from more than one bibliographical database, these duplicated articles were automatically excluded from the study selection.

2.3.1.2- Search terms

scientists	publish	code	Computer science papers	reproducibility
researchers	publication publishing dissemination disseminating disseminate	research Programs scripts	Scientific papers Experiments papers	reproducible

Table 2.1 Search terms and synonyms

The previous table shows the search terms that were used to find relevant studies. These terms were conjunct used together to determine the scope of the literature included. The phrases “code publication”, “publishing code in scientific papers”, “code in computer science papers”, “encouraging scientists to publish their code”, “encouraging researchers to disseminate their code”, “code dissemination”, “disseminating code in scientific papers”, “publishing

experiment's code", "publishing program scripts in academic papers", "publishing program scripts in scientific papers", "disseminate program scripts", "reproducible research" and "research reproducibility" were used to search the previously mentioned databases to retrieve papers relevant to the research area.

It was noticeable that these phrases led to other interesting research areas, such as "free software", "open source software", and "software sharing". However, such results are too general to answer the research question and are not relevant to this research.

2.3.1.3- Studies selection

In an attempt to sort the retrieved studies, inclusion criteria were identified to select appropriate studies to be included in this work. Studies had to: (1) be written in the English language; (2) explicitly discuss code publishing and address related issues; (3) explicitly address the theory of reproducibility in terms of code publishing; (4) address the scientists' views regarding code publishing, their reasons to publish their code or not, their motivation, and the barriers that they faced through this experience; and, (5) should not be duplicated, that is if a similar text is found in other publications then, one of these studies was considered in the review.

To determine the collected publications to be included in this review, each must satisfy at least one of these criteria in addition to the first one (i.e. be written in the English language), otherwise it was excluded.

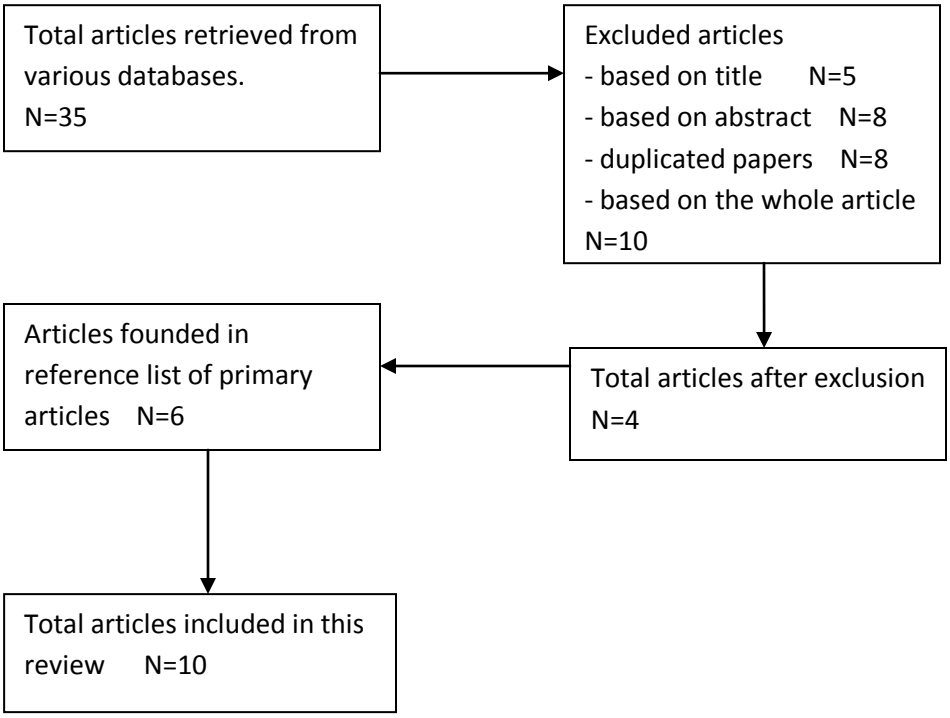

Figure 2.1 study selection strategy used in this systematic review

Figure 2.1 illustrates the process that was followed to select relevant studies from a total of 35 publications. As can be seen, there were initially 35 different publications retrieved from electronic databases. After applying predefined inclusion criteria on these publications, 31 were excluded after reviewing their titles and abstracts, the duplicated studies were also removed. Additionally, six sources have been added to the set of primary articles after searching their reference lists, which leads to a total of 10 studies.

2.4- Study quality assessment

To assess the quality of each primary study, a quality assessment checklist has been established in order to scale every individual study based on its comprehensiveness for the research topic. Table2.2 shows the quality assessment checklist that was used to evaluate each publication and the number of sources that met these criteria.

Number	Question	NO. of Sources
1	Did the investigation/analysis use any survey methods to address scientists view regarding withholding/publishing their code?	2
2	Did the investigation/analysis discuss the barriers that prevent scientists to publish their code?	3
3	Did the investigation/analysis discuss reasons behind the lack of interest that scientists show regarding code publishing?	3
4	Did the investigation/analysis discuss the benefits gained from code availability?	2
5	Did the investigation/analysis discuss the impact of withholding experiments details such as data and code?	3
6	Did the investigation/analysis suggest any solutions/recommendations to facilitate publishing process?	5
7	Did the investigation/analysis discuss the importance of code publishing in terms of research reproducibility?	6
8	Did the investigation/analysis embed into the context any previous research/study?	5

Table 2.2 Quality assessment checklist

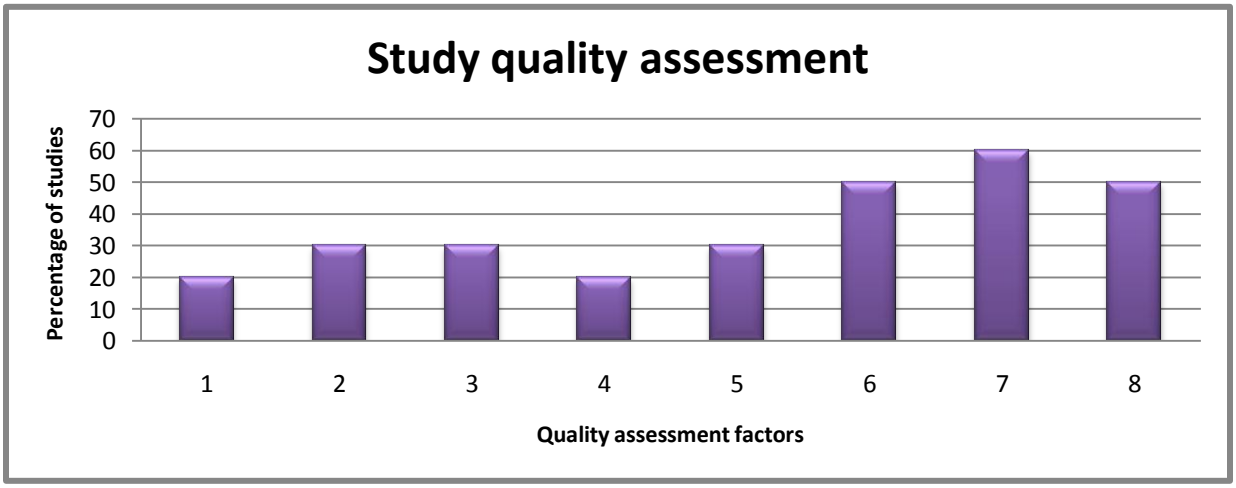

Figure 2.2 Percentage of publications that fulfill each quality assessment factor

Figure2.2 shows that only 20% of the total of these 10 publications have used a survey method to address the scientists' views regarding withholding their code. Furthermore, only 30% of the publications explicitly discussed the reasons behind the scientists' decision to not share their experiments' code alongside their published papers, even though almost 50% of them have suggested various recommendations regarding this issue. In addition, 60% of the selected studies explicitly paired the research reproducibility theorem with the ability of reproduce computational experiments and the code availability to achieve such an aim.

2.5- Results

Making the experiments' details (such as source code, data values, run environments, and system platform) freely available to be downloaded plays an essential role in the ability of scientists and researchers to reproduce experiments. Clearly, publishing code alongside published papers is significant to make the work falsifiable, and thus increase the scientific value of such work. Furthermore, code availability can facilitate the validation process by simply re-implementing the code (Vaughan et al., 2009).

2.5.1- Reasons for withholding experiments' code

Vaughan et al. (2009) have stated that a major reason behind the lack of interest that scientists show regarding publishing an experiment's code alongside their academic papers is that they consider the process of preparing the code for publishing as an overhead and extra work, while improving the code's quality is a time consuming task. In addition, some authors withheld their code because of competitive advantage purposes: by publishing their code to the public, some scientists feel that they will give others the opportunity to compete with them. To overcome this problem, Vaughan et al. (2009) have suggested protecting the ideas by using patents and licensing to protect their intellectual property before publication.

Through his investigation, LeVeque (2012) has noted that a major group of scientists have resisted the idea of releasing source code, and use several reasons to justify this conviction. For example, one of their excuses is that "the code was written for research purposes and not for public use". Another reason is that "it is embarrassing to publish ugly messy code". In response,

LeVeque (2012) has asserted that scientists must clean up their code to be suitable for publication. Actually, he believes that during the cleaning process, many code bugs could be discovered and fixed, thus the code quality will significantly increase. Another reason is that "the code may suitable to run only on a certain platform and does not fit elsewhere", this reason is quite reasonable since some code needs special hardware and software configurations, and only runs on certain operating systems. Despite this, LeVeque (2012) argued that it is worthy and more valuable if the code was available to be reviewed, even if it can no longer be run. Two effective solutions to the problem of system compatibility were proposed by LeVeque (2012): one is to use virtual machines to archive operating systems alongside the code; the other solution is to use code hosting websites that allow archived code to be run without the need for software installation.

In 2003, Thimbleby published a scientific paper that discussed issues related to publishing in computer science. In his argument, Thimbleby (2003) reported several reasons for scientists not to publish code. Documenting a system's requirements is one of these reasons, since they need to edit the code to meet these requirements, which is considered a time-consuming process. Another reason that makes code editing cumbersome is documentation. For example, the names of the variable may need to be changed and comment lines may need to be added to make the code more readable. At a commercial level, Thimbleby (2003) believe that there are close relationships between computer science and business. In some cases, the code of commercial scientific applications, such as software released by Microsoft, are sensitive. If their code were to be released, then the product would become unprofitable. Furthermore, software companies prefer to sell software upgrades, if they released their source code then they would have to take responsibility for fixing any problems. "Not just programs, but data too": Thimbleby (2003) has asserted that even the data which were used to test the code should be made available. This has proved to be useful in several cases, such as detecting semantic errors when translating code from one programming language to another.

2.5.2- Benefits of making code publicly available

Vaughan et al. (2009) have stated that various benefits could be gained by publishing code; one of these benefits is enhancing the research quality. Expecting that their code will be reviewed and probably used encourages scientists to write better code and improve the quality of their scientific contribution, which will significantly increase their reputation and success rate.

Stodden et al. (2012) have claimed that better research will be produced by practicing openness in science because scientists know that all of the details of their results will be publicly available to be tested and verified. Furthermore, for the authors themselves, by making such details available they can recreate all of the decisions that were made during the research. At the publishing level, code release will facilitate journals editors' mission since referees and reviewers can have a better understanding of the submitted results.

2.5.3- Technical barriers scientists faces regarding code sharing

There are several technical barriers that prevent scientists from publishing their code. According to Ince et al. (2012), these barriers include the lack of tools that are used to package the experiments' details in the research papers, the lack of scientific code repositories, the lack of awareness of the problems raised with code description, and the scientists' culture that states releasing code is an additional burdensome effort.

Another essential barrier, as Peng (2011) stated, is the lack of an infrastructure that allows scientists to exchange their reproducible work. Even though some journals now have their own online repositories, Peng (2011) found that these repositories do not support basic services, such as data indexing and searching data by indices.

According to a survey conducted in 2009, a group of 723 academic researchers in the machine learning field were asked for the reasons behind their decision to share or withhold their experiments' details, such as code and data. For 77%, the first reason was the time wasted in documenting and cleaning up the code. The second most popular reason (at 52%) was answering users' questions. In addition, for 34% the reason was copyrights. While 40% gave the possibility of applying for patents as a strong reason. For 30% of the researchers the reason was

the advantage that their competitors may gain through code disclosure. Stodden et al. (2012) have argued against this by emphasizing that this competition is likely to occur even if the data and code were not provided. In fact, from another point of view, releasing code can significantly increase the citation rates for that publication, thus positively influencing the author's reputation.

2.5.4- Code sharing in terms of encouraging reproducibility in scientific research

Today, it is clear that the rapid advance of computational science has led to a new form of inaccessibility since it is impossible to include every single detail of a scientific experiment in a published paper. However, not releasing the actual source code by which the main finding of a scientific paper would be reproduced is considered a major obstacle in terms of reproducibility, because any attempt to replicate the experiment without the code will fail Ince et al. (2012).

Stodden et al. (2012) have stated that in computational sciences, replicating or reproducing results in a presented paper in a conference or even in a scientific journal is almost an impossible mission. They added that this crisis is creating a growing gap. However, they argued that this problem could be addressed by practicing reproducible research.

Research reproducibility is a relatively novel concept in computational science. The idea behind this concept is that scientific publishing is not only about publishing papers but also about the data, code, parameter values, and anything else that facilitates the process of "reproducing" the findings of experiments. In the 1980s a scientist called Donald E. Knuth introduced a new notion: "Literate Programming". His idea was to produce programs with a literate description, not just lines of comments attached with the code. Knuth emphasized the necessity to be able to extract the actual code from the programs' "literary description". Afterwards, an earth scientist, John Claerbout at Stanford University, realized the importance of an experiment's computational details. Based on this concept, he introduced a system which enables researchers to combine code and data alongside their published papers. Claerbout had a conviction that "An article about computational science in a scientific publication is not the scholarship itself; it is merely advertising of the scholarship. The actual scholarship is the complete software development environment and the complete set of instructions which

generated the figures" (Kovac'evic, 2007). Claerbout's colleagues, Buckheit and Donoho have defined the requirements which are needed to introduce a reproducible research, such as "Anything in a scientific paper should be reproducible by the reader". In the early 1990s, David Donoho, who is a professor of statistics at Stanford University, started to practice Claerbout's ideas. He adapted Claerbout's claims to produce his "Wavelab" package. Wavelab is a Matlab based tool that could be freely installed as a toolbox. It was the first tool designed especially for reproducing results in several scientific papers (Kovac'evic, 2007 and Stodden et al., 2012).

LeVeque (2012) asserted his argument about sharing computer code by stating that the style that is used to present scientific knowledge has an essential role in assessing how reliable and credible the work is. In addition, LeVeque (2012) emphasized the importance of the reproducibility theorem and asserted that making the computer code that is used in scientific experiments available is an efficient way to ensure the success of the reproducibility process.

According to Thimbleby (2003), from a computer science perspective, good science is reproducible: reproducible means that all claims must be interpreted clearly and accurately so that others can build on these findings with the minimum effort. Thimbleby (2003) has suggested that the code does not need to be fully published; instead the authors can publish key pieces of the code that enable the readers to understand what has been done. The culture in computer science literature is that extreme disclosure of experiments' code is not required or expected since the idea is perfectly clear. This culture is also prevalent in other scientific fields. Thimbleby (2003) has asserted that the scientific community should change this culture because code availability is necessary for testing and validation processes.

Recently, researchers in several scientific fields have started to strongly call for the principle of reproducible research to assess the value of scientific claims. Figure 2.3 illustrates that reproducibility can range between full replication and no replication, depending on the degree of data and code availability. One of the critical obstacles that impedes the process of reproducibility is the absence of code (Peng 2011).

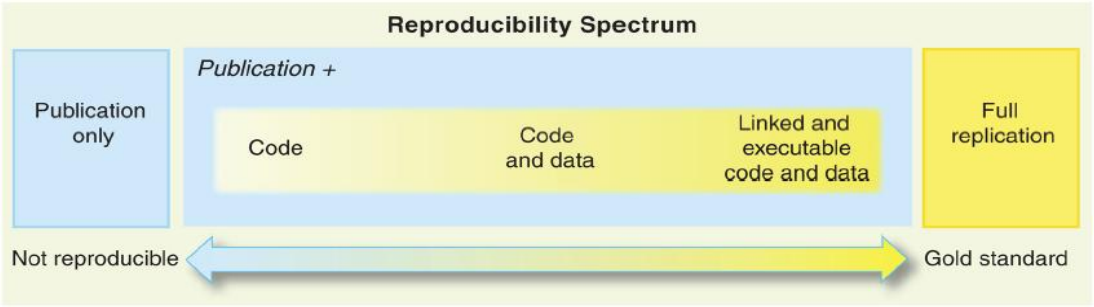

Figure 2.3 Reproducibility spectrum as Peng (2011) stated

2.5.5- Encouraging scientists to start practicing openness in science

Regarding encouraging code publishing, Vaughan et al. (2009) proposed several recommendations to change the scientific publishing culture. One of the easiest ways to achieve this aim is to assemble all the details in an archive file and use a revision control system, which generates a unique digital signature by using a "cryptographic hash function". This file is then published on a trusted host on the Internet and its URL is provided in the published paper. At the organizational level, publishers could refuse to publish unless the code is provided. Moreover, conferences could offer a prize for the "best" published code in terms of quality as (Vaughan et al., 2009) have suggested.

To encourage scientists to release their code, some journals have already started to change their publishing policies to achieve this goal. For instance, ***Science*** requires code to be mandatorily supplied by authors. By contrast, ***Nature*** journal requires a detailed description of the code rather than the actual code, which enables others to build their own code to establish a similar experiment. The problem of program description is that it could be interpreted differently at a syntactic or semantic level, which would lead to different results. To address this issue, Ince et al. (2012) have suggested expressing the description mathematically, or augmenting the code with mathematical expressions.

Peng (2011) has emphasized that journal reviewers should constantly request code to be submitted until this becomes a routine. He presented another example of efforts that made by the ***Biostatistics*** journal to encourage scientists to start practicing openness in their publishing.

Biostatistics has already introduced a policy by which researchers have been encouraged to make their experiments reproducible. Authors whose papers are accepted are asked to ensure that all of their code and data are made available online. Moreover, they can request a "reproducibility review". In this review, the referees run the experiment's code on the submitted data to ensure that the same results are generated. In addition, papers with submitted code receives a "C" mark and papers with data receives "D" mark, while papers that successfully pass the "reproducibility review" receive an "R" mark. ***Biostatistics*** started this strategy in 2009. By the middle of 2011, 21 from a total of 125 papers have received a mark; five received an "R" mark. Receiving an "R" does not guarantee quality; it simply means that same scientific claims were reproduced.

As a long-term recommendation, Peng (2011) suggested creating a universal center in which scientists in all scientific fields can place all their reproducible work and exchange their experiences.

At the Yale school roundtable in November 2009, scientists, journal editors, lawyers, and funding sponsors met to discuss the crisis of a lack of credibility due to a reproducible research shortage in scientific fields. This roundtable was inspired by the "genome research community", who has called for openness in genome sequence data. The aim of this meeting was to publish a document that formulates a set of recommendations to address data and code sharing in scientific publications. The roundtable participants stated that the lack of interest that scientists show regarding exchanging the details of their experiments forms a growing gap. They pointed to the fact that most computational experiments presented in conferences and scientific journals are unverifiable due to the absence of details. During the meeting the participants emphasized the importance of transparency in computational science since it allows researchers to build on previously achieved results, and they called for pressure to disseminate this culture. In terms of the scientists' role, the participants proposed six steps for scientists to facilitate reproducible research production, as follows. Recommendation one: when a scientist publishes a paper, all of the experiment's details (such as code, data, and statistical analysis) should be provided. This data could be placed on a university web page or on any code sharing

website, such as SourceForge and GitHub. Recommendation two: a version control system should be used to specify a unique ID to each code version, this ID will be changed whenever code changes. This identifier is used to track code versions. Recommendation three: include a description of the experiment's environment and the software version that was used to find the results. Moreover, a virtual machine image could be attached to avoid portability problems. Recommendation four: code licensing should be used, which is recommended by the "reproducible research standard" to protect intellectual property rights. Recommendation five: published papers and their preprints should be made publicly available on sites such as arXiv.org and PubMed Central. Recommendation six: publishing the nonproprietary copy of code and data should be encouraged wherever possible. In terms of journals' role, the participants identified several recommendations, which would help to ensure that their publications to be reproducible, as follow. First, policies should be established to encourage the provision of stabilized code and data URLs, these URLs are to be placed on the journal's website. Second, to facilitate the reviewers' mission, journals should provide servers which the researchers can use to upload the details of their experiments. Finally, details such as code and data citation should be required; this could be achieved by using standardized citation mechanisms, such as Data Cite (http://thedata.org/citation/tech) Stodden et al. (2010).

Ince et al. (2012) have also suggested several solutions. For example, research funding organizations should provide tools and code repositories that enable scientists to integrate all research data alongside their publications. At the educational level, scientific departments should strengthen the concept of reproducibility in their research activities to become a community culture.

2.5.6- Some case studies

Kovac'evic (2007) has informed a non reproducible research case study. He chose 15 papers that were published in IEEE Transactions on Image Processing; the papers were read and scaled between 0, .5 and 1 based on two criteria: experimental setup and reproducible research. In terms of reproducible research, code availability was one of the factors that were used to rate the papers. The papers and the authors' websites were searched to discover whether or not the

code was there. The results showed that none of the reviewed papers had code available (see table 2.3). Kovac'evic concluded his study by rating his own publications: he got 0 for code availability criteria.

Algorithm and Experimental Setup [%]					Reproducible Research Criteria [%]				
Algorithm details	Data details	Data size	Parameter details	Comparisons	Block diagram	Pseudo code	Data available	Code available	Proof available
80	33	46	46	26	0	60	33	0	100

Table2.3 The case study conducted by Kovac'evic (2007) on 15 papers published in IEEE Transactions on Image Processing

In 2011, Grubb et al. conducted an empirical study to explore several aspects of academic publishing, such as sharing data, reproducibility, credibility and effectiveness of peer review. This study aimed to make scientific findings publicly accessible. Using an open ended questionnaire, this study examined whether there is a consensus among scientists regarding the meaning of such aspects and if mechanisms were applied to achieve these concepts. The questionnaire consisted of 20 questions and was available in two workshops that discussed the interaction between science and the public. The first workshop was a two day conference held in California that discussed science and humanity, the other was a one day conference held in Toronto that focused on software design for computational scientists. The targeted sample was scientists who have had previous initiatives in open science. Questions were asked about data sharing, such as the analysis code, questions about the target audience of the published data, under what circumstances such data should be released, and the timing of publishing. Consistent opinions were found: the participants agreed that experiment data should be freely provided to the public. However, there was disagreement on the proper time for releasing the data: the answers ranged between “after review”, “as soon as possible”, “after publication” and “within reason”. Based on the participants’ response, Grubb et al. (2011) have categorized scientists into four groups: scientists who publish their data immediately; scientists who publish data eventually; scientists who believe in the necessity of publishing data but who face several

limitations (such as timing, patents, publisher restrictions, and system configuration); and, scientists who do not believe in data sharing.

2.6- Discussion

It seems that publishing an experiment's details, such as data, system platform, and code is an important aspect in scientific fields. As seen, from the reported studies that there is an urgent call for openness in the computational sciences and a change in publishing culture to fill the creditability gap caused by withholding this information.

Through surveying the literature and from the ten selected studies, it was noticeable that only 20% of the total 10 publications have used a survey method to address the scientists' view regarding withholding their code. Furthermore, only 30 publications explicitly discussed the reasons behind the scientists' decision not to share their experiments' code alongside their published papers, even though almost 50% had provided several suggestions regarding this issue. In addition, 60% of the selected studies explicitly paired the research reproducibility theorem with the ability to reproduce computational experiments and the code availability to achieve this aim. From these numbers, it can be seen that there is an urgent demand to provide the scientific community with a series of valuable recommendations to encourage them to share their experiments' code and to overcome the obstacles that they face in doing so, since removing these obstacles is crucial and has obviously not yet been adequately addressed.

This review investigates the reasons behind scientists' decision to withhold their experiments' code. Vaughan et al. (2009) have indicated several reasons that researchers report regarding this issue; the top was the time that was need to clean and prepare their code for publishing. Many scientists consider this task to be a time-wasting and time-consuming task; they prefer to work on new papers rather than waste time in improving the quality of their code. When Stodden et al. (2012) conducted a survey in 2009, 77% of the scientists who were asked also reported that the time needed to clean and prepare their code for publication was a major problem. LeVeque (2012) has argued against this reason and stated that cleaning the code helps the author to find bugs and fix errors. Copyright was another common reason that was reported to prevent the publication of code. Vaughan et al. (2009) have suggested some

solutions to overcome these issues, while in Stodden et al's (2012) paper there is no proposed solution to address these issues. Thimbleby's (2003) study was the only one that gave reasons from a commercial perspective and justified why business companies prefer to withhold their products' source code. Several studies have found that system configuration is an essential reason to withhold code. However, despite this, LeVeque (2012) has emphasized that making the code available to be reviewed is worthwhile, even if it cannot be run.

Although the papers reported for this review have introduced diverse purposes for hiding the details of experiments from a scientist's point of view, only a few papers have conducted surveys through questionnaires on a group of researchers to gain their perspective regarding code and data sharing (Grubb et al., 2011 and Stodden et al., 2012).

As seen previously, there are some barriers that impede the process of publishing code and thus prevent scientists from publishing code, even if they believe in the necessity of openness in computational science. Ince et al. (2012) was consistent with Peng (2011) in that the lack of scientific repositories and the lack of facilities in the existing repositories play an essential role in the absence of code from scientific publications. Peng (2011) went further and asserted that the lack of a universal infrastructure to facilitate the exchange of reproducible work makes scientists less eager to share the details of their findings.

Two of the selected papers indicated that publishing pseudo code as an alternative to release the source code was a problem; one of the main objections was that the misleading interpretation that can arise from pseudo code was seen as being able to lead to different results (Ince et al., 2012 and Thimbleby, 2003).

In this review, there were several studies that discussed code sharing in terms of encouraging reproducibility in computational science. "Reproducible work" means that there are enough details (such as code, data, and algorithms) by which a third party could independently recreate the fundamental finding. Kovac'evic (2007), Stodden et al. (2010), Peng (2011) and LeVeque (2012) emphasized the importance of transparency in scientific publication to verify how

valuable the scientific findings are. Moreover, making such details available increases the scientist's opportunity to build on other researchers' achievements instead of starting from scratch. In addition, they have agreed that code withholding is considered to be a critical barrier that impedes the reproducibility process. To assess how aware researchers regard this issue, a case study of 15 scientific papers was performed to check code availability; the results showed that none of the reviewed papers contained published code (Kovac'evic 2007).

Addressing code sharing issue, the scientific community has realized the urgent necessity of encouraging scientists to change their culture and start practicing openness in their publications, which will make scientific findings more valuable and falsifiable because it facilitates the validation process (Vaughan et al. 2009). In terms of citation acquisition, Kattge et al. (2014) have asserted that making an experiment's details publicly available will positively increase its reuse rate and thus it will receive more citations.

Several authors (e.g. Vaughan et al., 2009, Ince et al., 2012, and Peng, 2011) have emphasized the important role that scientific journals can play in encouraging scientists to release the code that they have used in their experiments. Several esteemed journals such as ***Biostatistics***, ***Nature*** and ***Science*** have already started to change their publishing policies to achieve openness in science**.** These journals have made code submission mandatory and they aim to make the process of cleaning code and preparing it to be published a routine and acceptable part of publishing a scientific paper. In contrast, Kattge et al. (2014) is convinced that restricting the acceptance of a paper to the release of code will not lead to a change in the scientific culture and effort should instead be made to change their attitudes at the first place.

2.7- Conclusion

During this review it was noticeable that in spite of the diversity of reasons provided in the selected papers to justify scientists' decision to not publish their code, the obstacles which they face, several suggestions to encourage them to practice openness, and the various benefits of sharing code, only a small minority of these investigations have supported their claims with a survey method (often a questionnaire) to develop a deeper understanding of these issues. In

addition, the lack of valuable recommendations was another obvious gap in the surveyed literature; for example, training sessions that may encourage scientists to change their publishing culture and facilitate the process of code publishing. This investigation has extends previous studies by conducting two interviews studies and a questionnaire survey. The first study was with a group of researchers who already have experience in code publishing to gain valuable insight into their views on this matter and what barriers that they faced. The second one was with the Research Software Engineers group to address these barriers in detail. At the end, a set of evidence-based recommendations have been proposed and evaluated to encourage openness in scientific research.

Chapter 3

3- Method

3.1- Introduction

The aim of this research is to provide evidence-based recommendations for overcoming the technical barriers that may deter scientists from publishing their code and thus encourage them to publish analysis code publically alongside their academic papers. As one of the main objectives of this research is to examine the scientists' attitudes regarding code publishing, a qualitative research approach has been applied. The results of interviews with software engineers regarding best practices in publishing analysis code were categorised using a thematic analysis to develop recommendations that were subsequently evaluated by scientists via a questionnaire.

The following section briefly introduces qualitative research and the particular methods that have been used in this project.

3.2- Qualitative research

In the literature, scientific research is categorised into two types based on the research question: quantitative research and qualitative research. If the appropriate answer to the research question is measuring how "much" something happens, then it is adequate to use quantitative methods. In contrast, if the research question is exploring how people behave regarding a certain issue or their points of view or experiences, researchers will use qualitative methods (Hancock, Windridge and Ockleford, 2007).

The literature provides several definitions for qualitative research. Nkwi *et al.* (2001) define qualitative research as follows: "Qualitative research involves any research that uses data that do not indicate ordinal values". It is noticeable that such a definition was based on the type of data used and generated by this kind of research. Speech, text, videos and audios are examples of these data.

According to Hancock, Windridge and Ockleford (2007), one of the reasons that qualitative research is criticised by the scientific community is that the sample group is quite small and its results cannot be generalised to a wider population. Qualitative researchers have justified this by stating that this subpopulation is the research focus and thus, generalising the findings to the general population is not the aim.

Qualitative research involves a systematic approach that begins by determining a research question that will guide the research process. To address this question, as a start, a literature review should be conducted; then, the research process is designed. During the research design phase, several decisions are made, including the type of data that should be collected, the techniques by which such data will be collected, what steps will be applied during the research process, what the targeted sample is and how the data could be analysed (Hancock, Windridge and Ockleford,2007).

3.2.1- Choosing the appropriate sample

The method of selecting the targeted sample for qualitative research differs from quantitative research. While in quantitative research, the aim is to generalise the findings at a statistical level as based on random sampling, qualitative research is based on strategic sampling. During the research design process, the researcher identifies a list of criteria that each sample should meet (Hancock, Windridge and Ockleford,2007).

3.2.2- Data collection techniques

For qualitative research, there are various data collection techniques, such as interviews, focus groups, observations and questionnaires. As each technique has its own strategy, purpose, benefits and limitations, researchers often combine more than one method within a single research study to collect valuable data for their analysis. For the purpose of this project, the interview method was adopted to gain in-depth information in relation to scientists and developers' views and experiences regarding the research topic (publishing experimental code alongside scientific papers). In addition, a questionnaire survey was used to obtain the

scientists' opinions regarding the suggested recommendations to examine whether such recommendations will encourage them to practice openness in their scientific publications.

3.2.2.1- Interviews

An interview is a process by which information regarding a social phenomenon can be obtained through human interaction (Ritchie and Lewis, 2003). Research interviews, as Gill *et al.* (2008) have claimed, are considered one of the most effective techniques, as they provide a deeper insight into participants' views, attitude and experiences in relation to a specific topic, and this is what makes it the most appropriate method to acquire detailed information from respondents. There are three main types of research interviews: structured, semi-structured and unstructured. Ritchie and Lewis (2003) have asserted that in qualitative research, the researcher is considered a research instrument. Therefore, to conduct a successful interview, the interviewer must have some significant skills, such as listening skills, a clear mind to respond quickly to what the respondents are saying and to formulate follow-up questions, good memory to note the necessary ideas that are mentioned at the earlier stage of the interview and curiosity to gain further details about what the participants said (Ritchie and Lewis, 2003).

In this research, semi-structured interviews have been conducted to collect the data. Within a semi-structured interview, a set of key questions is prepared to define the area to be discovered, but there is also the option of deviating from these questions when required (Gill *et al.*, 2008). Because of the nature of their questions, Gill *et al.* (2008) stated that semi-structured interviews provide the participants with the opportunity to explore the area of interest, allow them and the interviewer to diverge to follow-up on an idea in more detail and enable the respondents to elaborate on significant information that the researcher has never considered.

During this research, two stages of interviews have been conducted. The first stage of interviews was with scientists who are actively trying to publish their experimental code, they were conducted to uncover the obstacles they faced when publishing these codes. Then, the interviews' transcripts were thematically analysed to uncover barriers to publications which served as the basis for the second-stage interview questions which examined these issues and

potential solutions with members of the Research Software Engineering group within the university.

3.2.2.2- Questionnaires

A questionnaire consists of a set of questions that aims to examine humans' opinions and attitudes (McClure, 2002). The superiority of the questionnaire over other techniques comes from the fact that it can reach a wider sample of a population within different geographical locations. Regardless, Woods (2006) claimed that it is not a prominent technique in qualitative research. Two types of questions can be used within the questionnaire: open-ended and closed questions. Often, in qualitative research, the questionnaire consists of a mixture of the two types (Woods, 2006).

Closed questions provide a list of alternative options from which the participant can select the appropriate answer. Often, this type of question comes in the form of multiple choices, drop-down lists and checkboxes. Although, closed questions enable researchers to compare responses more easily, they are easier to process and analyse and they do not require much time to answer, they do not allow in-depth responses (Meadows, 2003). On the other hand, open-ended questions allow the participants to provide in-depth responses in their own words; thus, researchers can gain comprehensive meaningful information. The disadvantages of these questions stem from the fact that they require an additional effort from the respondents and they are difficult to compare and analyse (Meadows, 2003; McClure, 2002).

In this research, with the assistance of the Research Software Engineering group at the University of Manchester, a set of recommendations has been developed to overcome the technical barriers that may impede scientists from publishing their code. A questionnaire has been used to evaluate these recommendations with scientists and to determine how effective they are perceived to be.

3.2.3- Qualitative research approaches

Because of the diversity of the research questions in qualitative research, several approaches can be adopted. Each approach determines a specific type of information to be collected and

analysed. Several qualitative approaches involve looking for common patterns or themes among the collected data. To select the proper data collection technique and qualitative method, Guest *et al.* (2012) emphasised that it is essential to determine which dimension of the participants' experiences the research aims to investigate.

In this project, an inductive thematic analysis was used to analyse the collected data. This involves identifying themes and code within the collected data using a data-driven approach (Guest *et al.*, 2012).

In qualitative research, there are some ethical concerns that need be taken into account. Such concerns include the sensitivity of the data collected from the participants, the participants' privacy and the sensitivity of the subject of the study itself Northway (2002).

Northway (2002) argued that each aspect of the qualitative research process has its own ethical implications. From determining what the research question is to selecting the appropriate sample and conducting the analysis and disseminating the findings, at each step, the researcher will confront a set of ethical issues that needs to be managed.

3.2.3.1- Thematic analysis

According to Braun and Clarke (2006), a thematic analysis is a qualitative approach that aims to identify patterns and themes among the collected data. It facilitates the process of establishing models of human attitudes, thoughts and experiences. Moreover, it can illustrate several aspects of the research question embedded in the interviews' data.

A thematic analysis can be used widely to address the various types of research questions, from questions that examine the humans' views, attitudes and opinions to questions about the representation of particular social or psychological phenomena (Clarke and Braun, 2014).

Braun and Clarke (2006) have identified a theme as something that captures a significant meaning in relation to the research question. They claim that in this type of analysis, there is no right or wrong answer in relation to identifying themes; the most important thing is that the way in which such themes are created is clear and consistent.

Regarding the level at which themes could be identified, Braun and Clarke (2006) have said there are two possible levels: semantic or explicit level and latent or interpretative level. At the semantic level, the researcher does not need to dive into the data to discover what is beyond the participants' responses; he or she simply uses the explicit meaning of the data. At this level, the data are organised to present the patterns in a semantic way to emphasise the significance of such patterns to the research. In contrast, at the latent level, the researcher goes further beyond the explicit content of the data to examine the implicit ideas and assumptions therein. For the purpose of this research, the semantic approach has been adopted.

3.2.3.2- Starting the analysis process

The analysis process starts at an early stage of the research; it can be started during the data collection process, when the researcher starts to notice frequent patterns within the collected data. It is worth mentioning that some phases of this process are common to other qualitative approaches, so the following stages are not unique to this analysis (Braun and Clarke, 2006).

The first step is to become familiarised with the data. This could be achieved by reading and rereading the data carefully in an active way in an attempt to identify some common patterns. Before starting the second phase, Braun and Clarke (2006) suggested that the researcher must read the data at least once to develop an idea of what the code and patterns would look like. If the data are in a verbal form, they will need to be transcribed into a text form before conducting the analysis. Despite the fact that this task seems time consuming, Braun and Clarke (2006) considered it an excellent chance for the researcher to begin familiarising him or herself with the data.

Joffe (2012) has stated that the second phase is creating a code frame. After reading and rereading the data carefully, something called a code book or code manual is established to guide the analysis process. This book consists of a set of code that was derived from the collected data and the literature. As researchers cannot rely on standard categories, Joffe (2012) has claimed that devising a frame is a time-consuming task.

Once the code book has been created, the researcher must check its reliability. Reliability or inter-rater reliability is a measurement by which the consistency of the data that were coded by two independent coders is checked; the check determines whether the two coders could reach the same results of the coding procedure (Stemler, 2001).

This could be accomplished by applying the code to 10–20% of the collected data by two different coders and then estimating the correspondence between them. Therefore, the coders should code the same portion of data independently and calculate the degree of agreement between their results. If there is inconsistency, then a new code frame is developed with care taken to increase the consistency and transparency of that code frame (Joffe, 2012).

One method to calculate this reliability is by using Cohen's Kappa, which is a statistical method used to measure the degree of consistency of a set of coded data. The Kappa value can range from 1, which indicates perfect agreement, to 0, which indicates there was no agreement except what would be expected by chance (Hruschka *et al*., 2004). The Kappa factor can be computed as:

$$K = \frac{\mathbb{P}\mathcal{A} - \mathbb{P}c}{1 - \mathbb{P}c}$$

Where $\mathbb{P}\mathcal{A}$ is the proportion of data on which the coders agreed, and $\mathbb{P}c$ is the proportion whose agreement was expected by chance (Stemler, 2001). According to Landis and Koch (1977), the values of Kappa can be interpreted as follows: .81–1: almost perfect, .61–.80: substantial, .41–.60: moderate, .21–.40: fair, 0.00–.20: slight, <0: poor. If the Kappa's value indicated inconsistency, the code frame will be revised to identify the problems and propose clarifications. After that, a different proportion of the data will be distributed to the coders to be coded based on the edited code frame. This process will be repeated until a satisfactory agreement is reached (Hruschka *et al*., 2004).Due to the time constraints, the fact that this project is for a master's degree and no other investigators were involved, this reliability checking could not be applied.

After the code book has been developed and its reliability checked, this code frame will be applied to all collected data. During this process, the data chunks will be categorised into different code to facilitate further analysis. By the end of this step, the researcher can determine the number of interviews in which a particular code has occurred and how many times this specific code occurs within an interview, as well (Joffe, 2012).

Next, after all the data have been coded, these extracts will be associated with the pre-identified themes. This could be accomplished by analysing the list of different code to examine how such code could be combined to form a theme. In this phase, visual representations, such as tables, could be useful (Braun and Clarke, 2006). For each theme, a detailed analysis should be conducted by identifying the story that each theme should tell in relation to the research question.

3.2.3.3- Using a computer-aided qualitative data analysis tools

As qualitative research generates a massive amount of data that, if not organised carefully, could result in a real overload problem, researchers must keep track of all these data in an efficient manner (Kelle, 1997). In the mid-1980s, with the start of using personal computers, a group of qualitative researchers developed software packages to facilitate the process of analysing qualitative data. Tools do not have the ability to analyse data by themselves; they simply automated the processes of administrating and archiving huge amounts of data (Kelle, 1997).

This software helps researchers by enabling them to better deal with the vast amount of interviews data, assisting them in discovering patterns of code, comparing between groups and retrieving appropriate portions of data (Joffe, 2012).

Software packages, such as Nvivo and ATLAS, enable researchers to discover patterns among different themes within a range of interviews; moreover, such packages have effective functions through which the user can retrieve prevalent patterns of code in particular interviews, these patterns can be represented as charts, lists of text selections and visual networks(Joffe, 2012).

This research used NVivo, which is a qualitative analysis software package that allows the interviewees' transcripts to be organised and analysed to identify patterns and relationships among the data (Stanford University, 2011).

3.2.3.4- Disadvantages of thematic analysis

As an analysis approach, Braun and Clarke (2006) have claimed that a thematic analysis does not have the popularity that other approaches have within the research community, and this is because it is poorly documented and used. In addition, although the flexibility feature of the thematic analysis, which provides a various range of options regarding data analysis, is considered a significant advantage, in some cases, these options can be an issue, as they leave the researcher uncertain about the proper choice for his or her research.

3.2.4- Qualitative research approach advantages

According to Guest *et al.* (2012), one of the significant advantages of qualitative methods is that they offer the possibility of investigating participants' views and experiences more deeply as the research proceeds. During the interviews, the researcher has the ability to delve deeper and ask follow-up questions to obtain more detail. Another advantage of conducting interviews is that the researcher can have additional information that is not anticipated by the respondents. Furthermore, the researcher has the ability to ask the question in several different ways to ensure the interviewee fully understands.

3.2.5- Qualitative research limitations

Because the qualitative research has several stages (the collecting of data, transcribing of interviews, coding of data), the process of analysing such text is time consuming. Guest *et al.* (2012) have stated that a one-hour interview takes the researcher four hours minimum to transcribe from the recordings, and it takes two hours to read the text, several days to create the themes and one extra hour to code that text. Moreover, in the case of unstructured interview questions, it is quite difficult to perform a comparative analysis; it is difficult to establish a comparison between how different participants have responded to the same

question, because their responses can be varied for numerous reasons. In addition, the way the questions are asked can be an issue in this type of interview.

Chapter 4

4- Investigation of the barriers encountered by scientists trying to publish code

A series of interviews have been conducted with scientists who are publishing their code to investigate the reasons behind their decisions regarding whether to publish their analysis code alongside their results. These interviews were designed to uncover the factors, both cultural and technical; those affect decisions regarding publishing experimental code, and determine the potential obstacles that scientists may face if they intended to share their code.

The data for this stage were collected by interviewing eight researchers from different research areas: HCI, web interaction, visualisation of image data, text mining, data mining, HW/SW interaction and modular reasoning (see Appendix C for the sample details). One of them was not convinced about the necessity of code publishing thus he did not publish any code.

They have been asked about their views, the background issues and the changes that have convinced them to publish their code. In addition, the interviews have examined the code-publishing barriers from the interviewees' experiences, and they have been asked for their recommendations for other scientists in various fields to adopt practice openness (the detailed interview questions are given in Appendix A).

4.1- Coding

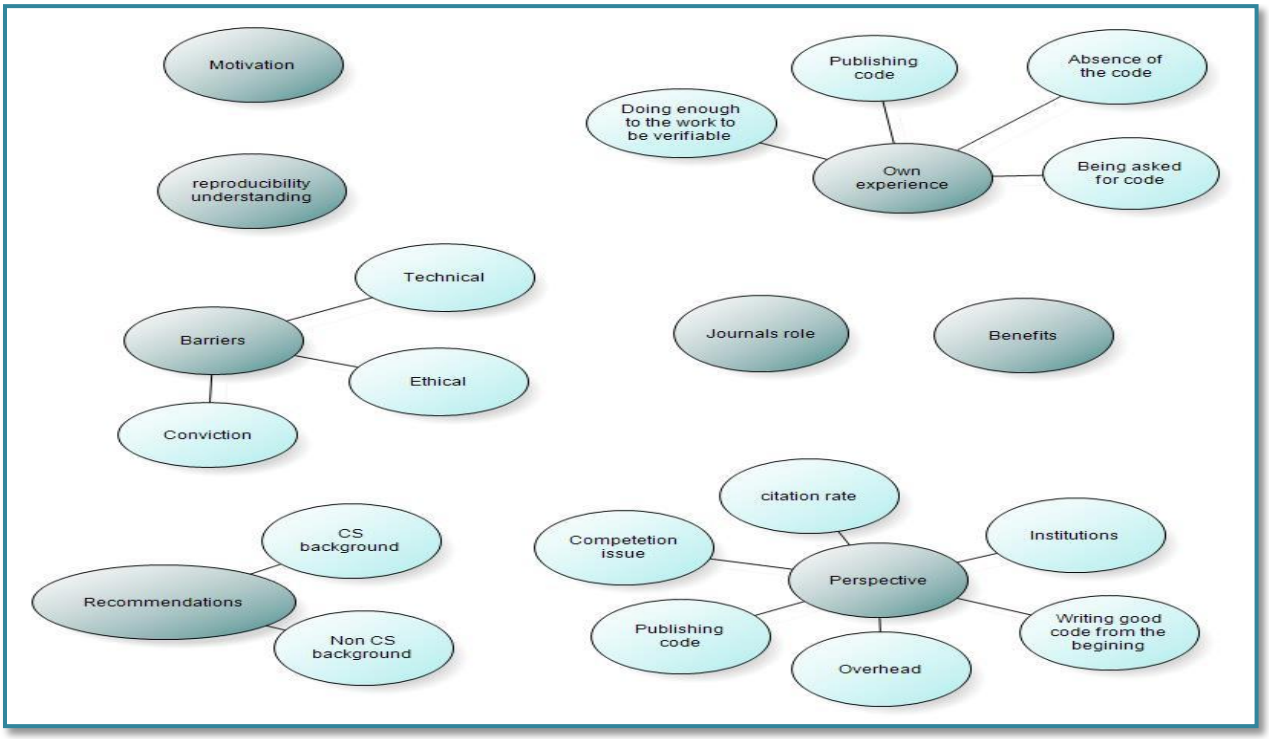

Figure 4.1 Themes and categories

Figure 4.1 illustrates the various themes that were used to segment the transcripts to smaller chunks of data. The eight main themes are: '**motivation**', which presents the motivation behind the decision of the interviewee to start releasing their analysis details, such as code, algorithms and datasets. The node '**reproducibility understanding'** contains their perspectives regarding the importance of the reproducibility of scientific research. The node '**benefits**' constitutes the various benefits scientists and readers can gain from publishing code. The node '**own experience**' describes the scientists' experience in several aspects in relation to code publishing. Node '**journals**' **role**' indicates the journals' role regarding this issue. The node '**barriers**' identifies the various types of barriers that scientists faced during code publishing. The node '**perspective**' consists of the scientists' views regarding several issues and the node '**recommendations**' contains of the scientists' recommendations to help other scientists publishing their code.

4.2- Results/discussion

4.2.1- Motivation

The motivations that convinced the interview participants to adopt the practice of openness in their publications were varied. There was a consensus on that by releasing code and other experimental details, scientists prove to the scientific community that all concepts they have introduced in their contributions were true; thus, the credibility of that work will significantly increase, which will positively enhance their reputation and their career. This theme was applied to four of the participants. Participant 1 said:

"I always thought that my tool had to be open, it had to be available, and everything I do should be available, trying to prove that you are not lying, trying to be open and say look I did this and even it is a small thing but it is true, and it is working"

It is worth mentioning that the interviewees have a strong conviction that making scientific work reproducible is part of good science, because scientists can prove the efficiency and credibility of their work, thus becoming accepted by the scientific community.

Participant 2 mentioned that even though exposing yourself to criticism is "scary", at the same time, and from a positive angle, it is a good way to verify your work so errors and bugs could be caught and fixed during the very early stages of that work before proceeding further.

Participant 4 pointed out that the University of Manchester has the credit of encouraging him to adopt the openness attitude in his scientific publications, as the reproducibility of the scientific research was a recently debated concept in the university.

"That's why at the work that I did here, I did everything I could in order to allow scientists at least to reproduce it with the same data, same code and same method and the same findings."

Participant 1 said that his supervisor during his PhD degree required him to publish all his details and make them available with only one click, and this requirement encouraged him to be organised at each step of his work.

Participant 6 emphasized that revealing experimental details is the only way to allow others to falsify their discoveries, unless the scientific work is only written assumptions, and all people can do is believing these assumptions, hoping they are true.

One interesting comment from participant 4 was that he has believed in the necessity of releasing all the details of scientific research since he was a child, and he was inspired by the Italian scientist *Galileo Galilee*, who taught him what the real scientific method should be.

"Since I was a child I was reading books about what is considered science, what Galileo did in order to teach us what a scientific method is, so one of the requirements for something that you publish for your hypotheses to be called scientific is the fact that it has to be a way for others to falsify your discoveries."

Regardless, Thimbleby (2003) claimed that withholding experimental code is the dominant culture in computer science. It is not expected or required to publish code, as the idea of that experiment is perfectly clear. From the previous transcripts, it was noticeable that there is a new trend among youth computer scientists regarding their code-publishing culture, as it is obvious this culture has started to change and the scientists have begun to adopt an entirely different conviction regarding this issue.

4.2.2- Their Perspective

4.2.2.1- Publishing code

Regarding the idea of releasing experimental code, this code was applied to five participants. Participant 2 mentioned that what makes him and other people "nervous", as he said, is the idea of being judged based on their work. Despite that, he argued for publishing the analysis details because it is a part of good science. Furthermore, in doing so, scientists can debug errors and mistakes in their work.

Participant 3 also pointed out the judgment issue; he claimed that the code he writes is not appropriate to be published. This is the reason behind his decision not to publish much of his work.

"Sometimes the code you develop is good for you but not good for others. Sometimes I feel embarrassed of my code and sometimes that's why I don't disseminate or publish a lot because of that. I am worry about how people will judge my code and I also think that if I want to publish something it should be assigned with comments, very well engineered."

Participant 4, who sometimes reviews papers for journals and conferences, emphasised that the availability of the code is always a priority. He said:

"One thing is I am always checking is the availability of code, if the code is published. I don't care if the code is open source, I just care of the fact that you have got this black box, even it is black, so I can push some button, I can install and try at least. I expect to have the same result; this is basic step nowadays."

A different perspective is provided from participant 5, who says that sometimes, when publishing complicated experimental designs that contain massive amounts of code, people will just use it and execute it to gain the same correct results; they will not even try to understand that material. Despite that, he insists on the idea of releasing experimental code publically.

Participant 7's opinion states that he will not reveal his code unless he will gain some financial benefits from providing it to others. He believes in the importance of releasing experimental code to the scientific community, but at the same time, he believes that business and science are so much entangled with each other; thus, if his code will not be used in a business project, he will not release that code. He also believes in publishing a pseudo rather than the original code, as a pseudo code is clearer, more abstract and can be implemented and optimised in any other language.

4.2.2.2- Publishing code and citation rate

Seven of the participants agreed that in terms of citation rate and whether there is any correlation between code releasing and the number of citations which the scientific paper could gain, the relation is weak and varies depending on the scientific field. They believe that in several situations, the scientific value is concentrated in the methodology itself, rather than in the actual source code.

Participant 6 has admitted that according to him, code availability is something that encourages him to cite the work; if the code is explicitly available, he will be happy to cite it, as he said.

Another factor plays an essential role here according to participant 2, which is the usefulness of the scientific work and how valuable it is; he said:

"I guess it depends on what you are doing, and how valuable people find it, if what you are doing is useful for the scientific community and you provide access to code and things which could be run and adapt, then in theory it is going to increase citations."

Participant 2 emphasised that increasing the number of citations must never be a goal; it will be a positive side effect of code publishing, but this benefit should not be an aim when a scientist starts thinking of revealing his experimental code.

Participant 3 asserted there is a direct relation between citation rate and tool availability instead of code availability. He claimed that scientists are gaining citations for their tools not code, and scientists look forward to inventing new things rather than repeating someone else's work, as replicating scientific work is infrequent and uncommon in the scientific community.

Despite what was mentioned in the background, Kattge *et al.* (2014) claimed that revealing the experimental details could positively influence the citation rates for a particular paper; the interviewees here believe these claims are not true in all cases.

4.2.2.3- Overhead

There was a consensus by six participants on the fact that the process of going back to the code, revising it and reformatting it in a way so that people can access it is not a priority for them. Participant 1 used the expression "insufficient loop" to describe this process.

Being selfish and preferring to publish a new contribution rather than going back to the code and trying to clean it up is an admission of participant 3; he also asserted that he would do this step only if it will enable him to publish another paper.

One interesting perspective from participant 4 says that if the person is a PhD student, then the cleaning up process is beneficial for him or her, as he or she needs to acquire several programming skills, and this could be accomplished by writing many code lines. He said:

"When you start forcing yourself to comment the code, using proper variables names, good object oriented schema of the code and things that we learned when we were in the first year of undergraduate. If you start doing that, maybe at the beginning you will have some difficulties, but then it will become automatically and at the end of your PhD, you will be a good developer."

He also thinks it is a great experience to receive messages from people saying that they installed your tool and they used your code in their experiments; with "crappy" code, this will never be experienced. Another perspective has been expressed by him regarding this issue, stating that:

"If you want to maximize the number of publications in the short term, then, it is a fair philosophy that instead of bothering yourself preparing code for publishing, you will publish a new paper. But if you want to have an impact on the community, the best way is to provide the community as much research as you can, so you can make others' job easier by using your methodology. There is a difference between short term and long term philosophy."

Participant 5 argues that it depends on whether this code will be used in other experiments. If the code will be reused again, the scientist should refine it to as perfect as can be. On the other hand, if the analysis code is only written to produce results, then it is an overhead and "he does not like to go back through this code again", he said; but at the same time, he believes that revisiting the code again is a good learning experience.

One way to make such extra work avoidable, as suggested by participant 6, is ensuring the code is well documented, up to date and well-engineered from the earliest writing stage. He stated that when working, he must work with scientists from different domains; thus, they need to understand and verify his code pieces. He is convinced that this approach is much "cheaper" than going back to the code after a while and trying to prepare it to be accessed by others.

From the literature and the interviews, it was noticeable that there is a general agreement on the fact that preparing the code to be publically accessed by others is overhead and a time-consuming task. It is unappealing to most scientists, and this fact is a significant reason that makes scientists think twice before deciding to publish any piece of code.

4.2.2.4- Writing good quality code from the beginning

This code was applied to three participants. Participant 1 asserted that he has this goal in mind, not to simply produce a good code but a well-documented code as well. The comments are not written to clarify the code to other people. His main audience is himself six months later, because he is convinced that after a period, it will be difficult to understand without these comments.

Participant 2 had the same opinion that writing well-crafted code is beneficial to him in the first place when he needs to go back through this code. He emphasised that he is already writing his code in a way so that other people would find it understandable and easy to access, because he is always keeping in mind that some people will read this code one day. He said:

"There is no point making messy code going to the end of the project and then think I have to spend more time going back through and sorting things out. You may just write good quality code from the beginning. It helps you change things quickly, and help you to analyze data quicker. So it not just a case that I am doing these to please someone else, actually it useful process to undertake."

Participant 3 has regretfully asserted that writing good code from the beginning is something that is always in his mind, but this kind of goal is easy to plan but difficult to apply for several reasons, time pressure being one of them.

4.2.2.5- Competition issue

In a survey conducted in 2009, it was stated that one of the reasons preventing scientists from publishing their code is the competition issue, particularly the concerns about being competed against by other scientists who are presenting their ideas in a way that is better than they did.

There was a consensus by five participants that this never prevented them from publishing; instead, participant 3 found this to be a good opportunity to produce a competitive atmosphere, which is good in science.

Regarding competition, participant 2 said that the possibility of this is weak, because, as he said, he will not publish his all details before publishing the paper; thus, no one can use his work before him.

In addition, regarding other scientists presenting his ideas better than he does, he emphasised that this is part of science and this is how science works. Actually, he admitted that he would be happy if someone admired his code or his data and used them in their contributions, because he thinks this will positively strengthen them and thus strengthen his career.

An interesting opinion that has been raised here is that this is not about advertising the work; the most important thing is that people can use, judge and improve the work. Participant 4 said:

"It does not matter how people talk about you, what matter is they are talking about you, so, I always prefer the fact that they can use, judge and improve what I did instead of taking care of the advertisement of it."

Participant 5 asserted that he thinks in empirical experimentations, competition is less likely to occur, which is why he does not have this kind of mentality regarding his own work, but he has it with others' works.

Participant 6 believes that the benefits that could be gained by sharing experimental details can overcome any concerns about the competition issue. He asserted the importance of scientific collaboration that can be accomplished by sharing code and data; this kind of sharing enables scientists to fix and improve their work, he said.

The collaboration issue has also been mentioned by participant 2; he said that trying to be secretive is unjust to science. In science, all scientists should have this sense of collaboration and should be eager to exchange their knowledge and ideas.

It is worth noting that competition was not an issue for this sample; it was not a reason deterring them from publishing their code.

4.2.2.6- Institutional services

The interview participants were asked whether they knew of some scientific institutions that offer services that may help them to improve the quality of their code, as well as what the possibility is to exploit these kinds of services and whether something like this will influence their decision to publish their code.

Participant 1 was uncertain about the idea that someone else is going through his code and trying to improve it without having previous knowledge of it. He justified:

"I don't know if they are going to comment the code, I don't know if their comments will be appropriate because it will be more related to the code."

However, he added that if he knew they were experts in his domain and knew how everything worked, then this would be interesting and he would not hesitate to use these services, he said.

Participant 2 found it difficult to answer because he comes from a computer science community and emphasised that if he was working in a different domain, these services would be useful to enhance the code's quality.

Participant 3 asserted that even though he would go through this experience, he would be embarrassed by the idea that someone else has revised his code; he does not prefer people to criticise him based on his code, he said.

There was a general agreement that these services will not affect scientists' convictions about sharing code, and they will share code regardless of these issues. These services may help by

reducing the time and effort that should be spent in preparing the code for publishing, but certainly, the absence of these services will not prevent them from publishing their code.

4.2.3- Own experience

4.2.3.1- Publishing code

The interviewees have been asked about their experience in publishing code. Even though they are convinced of the necessity for code sharing, some did not have the chance to publish full code in their previous publications for several reasons, but now, they are working on that in new publications. The code being unfinished is the reason that prevented participant 1 from publishing it completely, but he included pieces of the code in the published paper with a description of the full code, including what it is supposed to do and what results could be generated from it.

Participant 2 did not have the opportunity to publish his code due to the nature of his previous publications, which were about how certain aspects of medicine work, but the publication he is currently working on is experimental and he will publish the full code, he said.

Because many of the papers that he read were irreproducible, participant 6 was eager to include all his experimental details in his publications. He said:

"*I am very aware that a lot of the papers I read are completely irreproducible, very frustrated because you can't trust those results completely, you have to hope that they all tell the truth. I think I did enough through of the code I provided, the documentation that I wrote for the code, the comments in the code, the description of the algorithms, and the experiments and methods that people should be able to follow up.*"

Participant 5's perspective is that in terms of code publishing, he has attempted to publish all the details of his experiments, including the analysis code, datasets, results and even the plots. However, the problem was, as he explained, that all these pieces were distributed in several places; thus, if anyone wants to reproduce his or her work, he or she needs to assemble all such parts together to obtain one piece of reproducible work.

Participant 4 said that he completed all the required steps to make his work reproducible. He said:

"When I started my PhD, reproducibility was very important debated concept in this university, so this gave me the possibility to start thinking, and that's why at the work that I did here, I did everything that I could in order to allow scientists to reproduce it."

Even though he was convinced that publishing all the details is "the right thing", as he said, participant 7 was an exception; he decided to publish only the algorithm instead of the full code.

4.2.3.2- Doing enough to make the work verifiable

Regarding this issue, participant 1 said that he is always asking himself if he did enough and during his work, he attempts to make it as reproducible as possible. He added that he does this for himself in the future, not for others; he used the expression "in a selfish way" to explain that.

Participant 3 also asserted that he is constantly asking himself this question and the answer is no, definitely not enough. He justified this by emphasising that scientists always have priorities and always feel they are under publishing pressure. He feels embarrassed by his code, which is why in several cases, he chose not to publish many things.

An interesting perspective has been raised by participant 4, who says that asking this question is "a bit of the art". He considers himself "lucky" by being in the computer science field, because the process of preparing the scientific work to be reproducible is much easier than in other fields. He said:

"We are very fortunate that we are in this field. This means that reproducibility is much easier than medicine for example. All we can do is embedding our code and data in a virtual machine with all software installed so you will have ideally a big red button, so you just press that button to find same results that were published in the paper."

He also emphasised that although this process is technically easy, it is undeniable that it takes much time to be achieved; hence, there is a trade-off between the time the scientist needs to spend to allow reproducibility and the time he or she has to publish a new study.

For participant 6, this issue is very important and it is often in his mind when he writes his code, he said. By providing the full code, a well-written documentation, well-written comments and a full description of algorithms and methods, he thinks he tried his best to make his own work accessible and reproducible by others.

It is noticeable that the vast majority of participants have a sense of dissatisfaction in their efforts regarding this issue, and they are looking forward to their scientific work to reach higher levels of reproducibility, even though some have the impression that going through this process is time consuming.

4.2.3.3- Being asked for their code

All the participants have agreed that they have no problem with this at all, despite the fact that some have never experienced this situation. The difference was in the way they responded.

Participant 1 asserted that this is a strong reason that encourages him to go back through the code and modify it, because he knows someone else will see it.

Participant 3 went further and admitted that this situation may motivate him to write a "clean" code, as he said, but sometimes, because of the publishing pressure, he chose to "rush" it and complete it "badly".

"My response: my code is very messy you don't want to see it but I ended up cleaning it a little bit and sending it. Does something like this will be a motivation for me so start writing a clean code, yes, I have that in mind but you cannot always follow the plan you have."

Participant 4 emphasised that as well as publishing his code in repositories, all he is going to do is point others to that repository. He added that he would not do anything special; others can just go and download it, he said. He was not the only one with this perspective; participant 7

also asserted that if someone asked him for his code, he would send it immediately without "bothering" to try to clean it up. "I would not be a code shy", he described.

When asked for his code, Participant 6 said that he has sent the link directly, hoping to gain some citations from his code.

Participant 7 added that he would use the General Public License (GPL), which is used widely for licensing free software, to protect and preserve his rights before sending the link to the requested code.

4.2.3.4- Absence of the code

Five out of eight participants agreed that in some cases, the absence of the code in most scientific papers is a significant barrier that may impede scientists from performing extended research, thus affecting the flow of the scientific process negatively. Participant 5 said:

"The absence of the code is an obstacle for empirical work. I cannot set down and try to re-implement these people code, it is impossible. There is complex artifact that could take a year to re-implement it."

A personal experience was conveyed by participant 6 that supports the previous perspective, which is:

"I had this problem recently, there were three algorithms which were held up as standards in the domain I work in, and there were no implementation of them anywhere, so I had to write them from the scratch. Each algorithm took a week to develop and test to make sure it was correct, so that considerable time really, significant obstacle in terms of time, money as well."

Participants asserted the availability of the code is not only necessary for performing extended research, but also to clarify what has been done in the experiment and why it has been done. In addition to the absence of the code, respondents considered the lack of a deep explanation to be a significant barrier that could impede the extended research. Participant 2 said:

"When I read a lot of papers it is not always clear what they have done, and why they have done it, a lot of people don't say we did this to the data because of that, you just get: we run this test and we got these

results, it is hard sometimes for you to understand what they might done."

Another important aspect that has been raised was the availability of comments, which are considered guidance for the readers; they are as important as the code itself. Participant 3 told his own experience of a situation when the absence of comments was a significant obstacle when he attempted to test software.

It is undeniable that the absence of the code is an obstacle if someone wants to build upon a piece of research, but as a computer scientist, participant 4 emphasised "there is always another way to do it, you will always find an alternative way in order to overcome this problem", he said. He asserted that the absence of the code was never a problem, because he always finds it to be an opportunity to build the scientific work from scratch and thus gain a deep understanding of that work.

It is noteworthy that the previously mentioned opinions have been influenced by the fact that the respondents belong to the computer science field and have a programming perspective. It is possible that these views could be quite different if the sample was from another domain.

4.2.4- Journals' role regarding code publishing

The participants were asked whether providing the code was compulsory to be allowed to publish in a scientific journal. They all confirmed that they have never been asked for their code. It was noticeable that the respondents were divided into two groups based on what their attitudes were regarding this issue.

Some of them considered this request a strong incentive to make them eager to review the code and try to clean it up, fix bugs and make some improvements to be suitable for publishing. Participant 6 said:

"No one ever asked for a code. No doubt if there was a request to share the code directly by the journal I would look it again to make sure, and fix anything I am not happy with."

On the other hand, the other group had another perspective in that this request would make them search for another journal that does not demand code. People always prefer the easiest path, as participant 3 said:

"If a journal asked me that I need to submit code, I would choose another journal perhaps. I will choose other journal that doesn't have that many overhead and that demanding and this is not because I don't trust my code, but I will go to the easiest path."

Participant 4 has experience in reviewing papers for some journals. He asserted that even though reviewers do not often ask for the code, they are always convinced that not providing the code is a symptom of the fact that the research does not work well. He added:

"If your code is not published, it is very likely that it is not ready to be published at all. This means that even if the reviewer asked for the code and then rejects the paper for this reason is very likely that this guy will end up submitting the same paper to a different journal."

The previous transcripts have supported what Kattge *et al.* (2014) claimed about restricting the acceptance of a paper to the release of the code; Kattge *et al.* (2014) were convinced that forcing researchers to submit their code would not lead to a change in the culture. Instead, effort should be made to change their convictions in the first place.

Regarding the role of journals in encouraging scientists to release their code, participant 4 emphasised that recently, there has been a step towards encouraging scientific reproducibility by some journals. He explained:

"I am experiencing kind of increment of sensibility for these things so as reviewers, there is a change of policies so many times they ask us, did the guy provide the code, can you reproduce the same results, and can you reproduce the methodology and then replicate results."

4.2.5- Barriers

4.2.5.1- Technical barriers

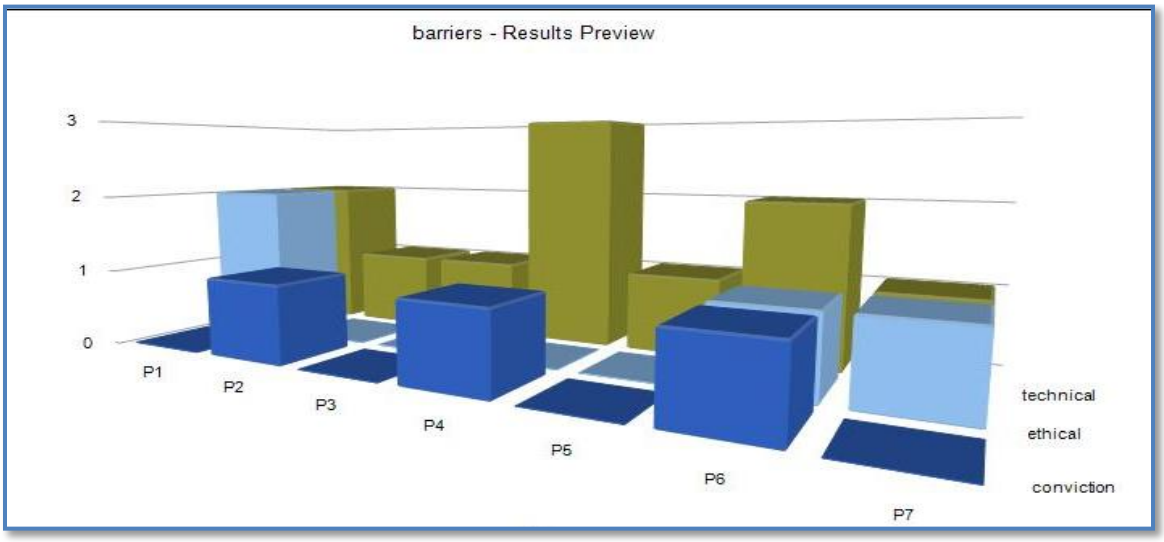

Figure 4.2 Different types of barriers

As shown in figure 4.2, it was noticeable that in terms of barriers that may impede publishing code efforts, technical barriers encompassed the largest portion of participants' responses, which indicates the importance of addressing these issues to fulfil the code-publishing process.

There was a consensus by seven of participants on the existence of technical obstacles that deter the publishing process in some cases.

One of these barriers was the problem of organising the experiment's related components in one place. For one experiment, there are several files, including the analysis code, the dataset, the analysis plots and the database in which the data could be stored. All these files must be placed in a single directory and they should be executed in a specific order. When the participants intended to publish their experimental details, they said they had to distribute these files to several places, which formed "islands of information", as participant 5 said, and these islands needed to be "stitched" together to enable others to execute and reproduce the experiment.

Another significant obstacle from participant 4's perspective is the space limitation of the webpage that is offered by the university. As he said, his webpage was restricted by a 200-megabyte limitation, which forced him to publish his details, the dataset and the machine learning models on an external server and to provide the link to that server in his published paper. He recounted his experience, saying:

"At some point I wanted to publish my pipeline, now in text mining both datasets and machine learning models can be quite heavy in term of space. What I wanted to publish was something like 2 gigabytes big which is not a lot, and technically I could not put it in my website because we have 200 megabytes limitation for websites."

The same participant mentioned another important issue for several scientists, including the lack of a central repository in which the code could be stored. Thus, related aspects, such as security, privacy and access rate monitoring, should all be automatically guaranteed by the university's IT department.

The difficulty of using online repositories that are widely used to publish experimental code is another issue that has been raised by five of the participants. They emphasised that dealing with these repositories is not straightforward and people need to be trained in how to use them, even people who have a technical background. Participant 3 said:

"It is not easy to use. basically because I used it for the last years, I had not try to learn it, I just was expecting it to be easy and intuitive but all this protocols and so on I found then very difficult, more than difficult, confusing because there are several commands and I am not very sure about the consequence of doing these things so I am very scared about using them."

Participant 4 believes these repositories need an "introduction" to know how they work and to become familiar with them. He said:

"It need a little bit of introduction in order to be used in a proper way, I found it a little bit hard in the beginning, and I spent two days looking for documentation. It was semantically difficult."

This opinion has been supported by participant 2

"People need some kind of training to know how to use repositories because they are not straightforward, especially if you don't have a technical background, pushing and pulling to these things and the terminologies that are used and so forth can be difficult for people. For non technical background, but I suppose even for CS people if they don't have experience to use repositories and version control ,anything like that they might find them difficult."

Participant 7 has gone further and tried to be more specific by identifying this difficulty; he believes these repositories have a sophisticated command that may be complicated for those who are from other scientific domains if they have not been trained in their usage. As a solution, he suggested:

"To make these tools a bit more full professional, I would say, you may have a web based interface on top of them saying, that communicates with it which makes it easier; so scientists need just to be able to do it in few clicks. Simple to use interface between the repository and the user may really help."

Participant 6 disagreed with this opinion, as he believes all these repositories are easy to use and people do not need to be trained to use them. He justified his perspective by asserting:

"In fact some of the scientists who I work with are physicist already use them, so it does not need really to have any significant technical background."

4.2.5.2- Ethical barriers

Three respondents have summarized the ethical barriers that they have faced into two main issues. The first is the privacy of the datasets they use in their experiments. Because of the nature of capturing these data for their work, and because such data belong to some people who always have concerns about privacy, disclosing these data is considered a privacy violation, which is illegal and may lead to legal sanctions. Participant 1 said:

"I would say the main barrier for the data is basically privacy, the fact that I can publish results, I can publish data that come from these results, but I can't publish their raw data, I am already because the nature

of the capturing, having many people asking about the privacy of the data."

Participant t 6 added:

"In terms of barriers, I faced data access barriers, I have accessed datasets which have been supplied by foreign governments, and foreign research agencies, although I have accessed to that data, I don't have permission to share them."

The other ethical issue is having publishing permissions from supervisors, funding agencies and project stakeholders. Participant 7 said:

"If you want to do that there is the supervisor factor, depending on what type of projects is yours, the funding agencies may give permission to do or may not so basically the stakeholders on your project have some sort of control on your project".

4.2.5.3- Conviction barriers

This barrier was mentioned by participant 4 who has the view that this kind of barrier is "more tricky", as he said, and thus harder to overcome. He explained:

"It is belonging to your philosophy of sharing, if the scientific process in your mind is ending with the publication or starts with publication, this is a big difference. Lots of people just care about getting their paper through the review process, once it is published, and then everybody is happy, while there is other people for which their view is that the publication is just the beginning. What happen after is that, people should be able to build on your discoveries so you are just the basic step for them."

To overcome this barrier, he asserted that efforts should be made at several levels. With regard to the university's role, he suggested stimulating students to think about publishing during the undergraduate stage, because in his opinion, it is always preferable to provide the true philosophy, as well as what the meaning of being open in science is and what types of problems this aspect can solve.

4.2.6- Recommendations

4.2.6.1- Recommendations for computer scientists

The first recommendation was provided by participant 2, who stated it would be useful for other scientists who have never had this experience before to gain closer insight into some situations where scientists exchange their knowledge. This would also be useful to determine how those people think regarding this issue, how this aspect could be beneficial for both sides and how this could be useful for himself when he or she uses somebody else's work. In doing so, he believes that these scientists will start rethinking and trying to release their code.

He also had another recommendation for the universities; he asked them to provide these kinds of scientific repositories to their staff and students at no cost and to simplify their usage as much as they can to be suitable for scientists from various domains. Furthermore, providing some training sessions and workshops on how people can use these tools would be helpful. He explained:

"Many people are thinking I have to go and find a repository, register, learn how to use it, push the code on. They might not want to go through these stages just to seek for sharing."

Participant 2 had a different but related recommendation. He advised other scientists when they intend to write a piece of code that they should write it in a way so it could be published and other people could read, verify and reuse it. It should be in a well-engineered format and well documented to avoid the overhead of going back through the code and trying to clean it up to an appropriate form; this overhead may sometimes be the reason behind their decision not to publish their code.

Another recommendation is related to the concerns about being criticised; participant 5 asserted that this issue must not prevent scientists from publishing, because he believes that "no one has a perfect code" and "it does not matter" if someone has judged them based on their code. From a positive angle, he emphasised that the feedback will significantly help them to enhance the code, fix bugs and thus strengthen their programming skills.

An interesting recommendation was provided by participant 6 and covered several important issues. He said:

"If you have a new idea and you have not publish yet then keep that to yourself, protect that but once it is published then share it as widely as possible. Use accessible standard libraries, don't use very rare programming languages, and try to use languages which are easy to get into, publish data in reusable format."

The last recommendation is helpful to increase the scientific work's credibility. This aim could be achieved by publishing the full code and providing the hardware configuration on which the experiment was run so people can verify the work; moreover, they can compare their work with the one that was published to generate the same results. This comparison process could be beneficial to the author, as well, because their work will be inspected deeply so any bugs and errors could be addressed.

4.2.6.2- Recommendations for other scientific domain scientists

These recommendations were provided to encourage scientists from other computational domains to release their analysis code.

Because scientists who do not have a technical background have a very limited grasp of programming, the first recommendation is to gain some training in using software engineering principles to make the code more usable, more accessible and more sharable. In addition, using computational techniques might help them to follow the software engineering practices in their field to refine and polish their code so they could be shared easily.

While contacting computer scientists and asking them to refine their code seems a proper solution, participant 3 asserted it is not a long-term solution, because at the end, any code needs "constant maintenance", so the scientist needs to have full control of his code; thus, delegating someone else to accomplish this task is inadequate.

Participant 5 asserted the importance of training these scientists in how to use electronic sharing tools, such as repositories, to facilitate their daily tasks. Participant 6 has the view that it should be the university's role and responsibility to train those scientists to deal with

computational methods. He said:

"The university role is to train them; they should have training in computational method. It will be helpful if the university did it. They could send students over computer science department to get them trained which will be helpful."

Participant 7 suggested that the services provided by the institutions mentioned earlier could play a significant role here, as those scientists have limited programming skills, such services will be helpful to refine and enhance their code.

From this set of recommendations, it is noticeable that the participants have tried to overcome the barriers that they faced during their publishing experiences. They covered several aspects that might be useful to be addressed in detail to help scientists release their code.

4.2.7- Code publishing benefits

Three of the interview participants emphasised that numerous benefits could be gained by releasing experimental code, not only for readers but also for the scientific community and the authors themselves.

Making the code available is a strong motivation to encourage scientists to develop good quality code, as knowing that the code will be verified and evaluated by other people makes the scientists eager to write good, reliable and readable code, as participant 3 said. Participant 2 had the view that practicing openness in scientific publications can positively increase the credibility of that work, thus improving the author's reputation and enhance his or her career. Participant 4 added:

"You get for free the fact that your paper is automatically advertized."

For the scientific community, one of several valuable advantages that could be obtained by code publishing is scientific collaboration, in which scientists exchange their significant findings, build upon each other's results and suggest some improvements that could enhance their scientific value. Participant 2 said:

"The bright light of the whole idea of this collaborative approach is that people will improve something incrementally and everyone can have an ownership of this"

He added that as scientists, they do not want to be in the situation where they are all working towards the same findings or developing the same ideas simply because that they want to "keep it secret"; it is "injustice" for science.

4.2.8- Reproducibility understanding

Participant 1 emphasised that to make the scientific contribution reproducible, making the code available is not enough. Effort should be made to ensure that such code work properly, because he experienced several situations in which "he has struggled because the code was there but not working, so he had to spend more time trying to fix it to see how it works".

As scientists, participant 2 asserted that they must make their scientific work as accessible as they can. He explained:

"It is very important as scientists to try to make our research as open, accessible as we can to others, because the ability to reproduce what someone done and follow the stages is a co-part of science."

Participant 6 was frustrated, as he said that when he entered the academic field, he found that the vast majority of scientific contributions were irreproducible. He believes that in some scientific fields, such as medicine, which are related to peoples' lives, it is critical for the contributions to be reproducible.

Participants 3 and 4 agreed that scientific reproducibility is a fundamental principle in science. Participant 4 emphasised that it is an important aspect for falsifiability in science, which is a significant requirement to ensure that scientific claims are verifiable, as it is a milestone telling others what has been done thus far.

It is worthy to note that the participants have that awareness about the importance of the reproducibility aspect in scientific research and they started to adopt this new attitude when they planned to publish any contribution. They have attempted to make their work as

accessible as they can by offering all the essential components to allow others to verify and reproduce their work.

4. 3- Conclusion

It is noticeable that this group of scientists has the awareness of the importance of scientific reproducibility to the scientific community. Because of such awareness, they attempted to make their scientific contributions as accessible as they could by providing the essential details of their experiments. Even though they belong to a technical field, they have confronted several technical barriers that in some cases impede the process of code publishing. To facilitate such a process among those scientists and scientists from other computational fields, these barriers will be addressed in more detail via a set of interviews with the Research Software Engineering group at the University of Manchester.

Chapter 5

5- Investigation into research IT professionals' views on publishing code

A set of interviews were conducted with seven engineers from the Research Software Engineering group at the University of Manchester to develop recommendations for overcoming the technical barriers that may prevent scientists from publishing their code. The seven participants are working in domains as diverse as IT services, as members of the myGrid team, and at the SSI (see Appendix C for the sample details).

5.1- Coding

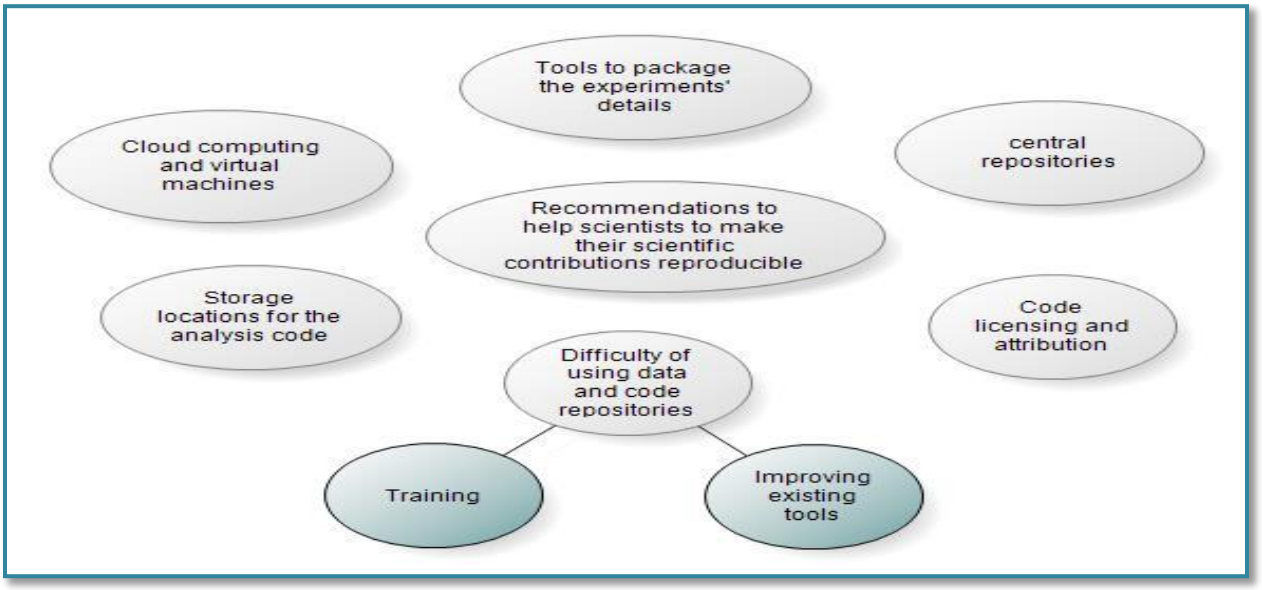

Figure 5.1 Themes and code which were used in this interview study

Figure 5.1 illustrates the themes and code that were used to categorise or "code" the interview data chunks using the process described in chapter three. Since during this stage, the focus was on the technical barriers that were identified by scientists during the first-stage interviews and mentioned in the literature, these themes were conducted from the previous interviews and

used as a basis for developing this stage interviews' questions. These barriers were to be addressed individually with the Research Software Engineering group with the goal of eliciting a set of useful solutions from those engineers to overcome them. In addition, the Research Software Engineering group was asked about some suggested solutions to examine whether they can contribute further to address this issue.

The interviewees were asked about using cloud computing and virtual machines to facilitate scientific reproducibility, the availability of places in which scientists can store their experimental details, the effectiveness of establishing an institutional repository that will enable scientists to publish all their details, the difficulty of using code repositories, code licensing issues and tools by which scientists can package the experimental details into one single file (the detailed questions for these interviews are in Appendix B).

5.2- Results/ discussion

5.2.1- Storage locations for the analysis code

One of the issues the scientists mentioned was that they do not have places to store their code; there are no sensible resources in which to contain their experimental details. There was a consensus among all seven participants that this is not a real issue, as the actual problem is the lack of awareness and education about the availability of such places. Participant 1 said

"I think a lot it comes out to education; there are places that you can put code. There are code repositories like GitHub, Zenodo and BitBucket"

He also emphasised that it is up to the university's research IT services to increase that awareness and give students proper advice by providing them with a list of the options that could be used for this purpose.

Participant 2, who is working for a scientific journal as well, asserted there is a big focus now on encouraging scientists to make their source code available. Many journals are pushing the authors of scientific papers to publish their data and code by providing databases that permanently host the data and code associated with scientific papers. He also emphasised that

scientists should be aware of the importance of good documentation regarding scientific reproducibility. He added

"Even if you put your source code in GitHub, you still may not be able to reproduce the results for number of reasons. the problem is that when I execute the code with the same datasets, I could not get the same results, the problem is that there is sometimes one of the tools, the way it processes the data, it is nondeterministic, so every time you run it, you will always get slightly different results because there is some kind of randomness, so you have to understand how the workflow, how the data analysis works. It is important that the data analysis should be well documented. Documentation is the important thing."

As an IT services member, participant 3 said that the university IT services offer several utilities for PhD students and researchers, but the problem is that she thinks such services are not well advertised and admits they need to increase awareness about their availability. She said

"IT services provides source code management services which offer central source code management via Github or Apache Subversion to support computer programming"

She added

"There are several places to put the code, when you log in with your user name and password, you get your P drive that can be quite small, sometimes 250 megabyte, but you can ask that to be expanded to 2 gigabytes. Again there is another thing that is not well advertised to them, when you log in again, with your P drive, there is a public HTML folder, you can put things in there, and that page will be on the university webpage, it is like a single page or two, it is tiny in comparison"

It is worth mentioning that the previously conducted interviews during stage one also supported this idea about the lack of advertising of such services, as no one from among those scientists has mentioned or even seems to know about their existence.

Regarding the limited webpage space issue, participant 1 argued that for this particular problem, there is always "a way out"; he suggested that scientists could easily "link outside". They can contain their data on an external server and link to it in their webpage.

Participant 4 emphasised that these tools are more likely to be suitable for software engineering issues rather than research. "If you are just in research, you probably don't know about these services, unless you actually writing software", he said.

Participant 5 has tried to go further and provided some effective solutions in terms of increasing scientists' awareness regarding the availability of these places, including how to exploit them effectively. She suggests:

"*I think there are places but I think a lot of researchers are not aware of them, they know a little bit but they don't know what to choose, they don't understand the benefits, they don't know the advantages so I think there should be some training in the early part of the research path. So, maybe at the PhD level, or maybe even earlier, the awareness should be raised. Also it would probably be good to have some clear comparison guidance because just raising awareness is not enough because I really suspect that a lot of them are aware, they have heard about some of them like GitHub, but they don't understand why it is better than others. I think it is a continuing process so I suppose some of IT support services could be those that would monitor the scene constantly.*"

5.2.2- Tools to package the experimental details

Another barrier that was mentioned by more than one participant in the first-stage interviews and that was mentioned in the literature is the lack of tools used to package the experimental details in a single directory: "the data are scattered among different islands", one participant said. Trying to address this issue, the Research Software Engineering group has been asked about their suggestions; how scientists can link or maintain links between different components.

"there is one activity in this group here within the university called research object.org, they got these kind of tools to help package up research, so you package up your methods, your data, your graphs and publications in a meta data format, so you can bundle and represent your research, and then you can explore that", participant 1 said. He emphasised that:

"It is difficult to have everything in a single file because the tool is very new, it is like ten or fifteen years ago, every one talk about it now it is anew thing, maybe ten years from now there will be some solutions where

people will have some tools."

Participant 2 has also praised the idea of this research object project and asserted they are looking to use it in his journal to relate all experimental details together. He said:

"The solutions I have seen is get these metadata frameworks, like research object, so these research objects contain like a bag of things that associates pieces of research, you can have that bag, it can be a set of links in a single package and then there will be relationships between the links."

As a member of the research object project team, participant 5 tried to explain the idea of it in more detail. She said:

"there is a research object, a sort of project that we work at here, this is supposed to be a mechanism that helps to do exactly what you have just said, so get the research object just like a bundle, and you can access the data, you can access the code if there is any code, the description of the pipeline if they have a pipeline, paper that associated with that, maybe some notes, so really everything you may need from the researcher experiment."

Since the fact that source code sometimes need to process terabytes of data, it is hard to package all these details in one single file and apply it to the available services. Participant 3 has suggested that services, such as GitHub, can be integrated with external research storage, such as Iceland, and then a read me page can be created at the top of the GitHub repository to point to the correct data source locations on the Iceland service.

Participant 4 claimed there are some tools that already exist through which scientists can submit a list of data files instead of a single file; then, these files could be linked together as a single package or single reference so they reference a single object that actually has a big list of files. He also claimed that the idea of packaging all the data of an experiment in a single Zip file is applicable, but participant 3 disagreed with this claim and emphasised that such an idea cannot be feasible as a long-term solution.

Participant 5 asserted that the focus should be on trying to exploit what already exists rather than inventing new tools to avoid a proliferation of standards that may confuse scientists. She had a different idea:

"I think some training in a generic level, so saying ok it does not actually matter that much whether the data in Figshare or maybe it is in MyGrid, but this is how you should describe your data, this is what should be metadata, and this is how we can make it available for everyone."

She added

"In fact details don't really matter that much, if you understand what Meta data is, you would be even able to describe all the experiment using the Meta data. I think workshops and these workshops should be kind of handouts, not just in a lecture style, it should be fairly practical."

Regarding this issue, participant 7 suggested that researchers could benefit from the Research Data Management Plans service, which is provided by the university IT services, to accomplish this goal; these plans consist of several representations of the data that are part of the research project, including how this kind of data could be stored, shared and preserved. Moreover, The IT service can assist in the development of these plans by offering general advice regarding several aspects, such as ethics, copyrights, licensing and external data storage.

5.2.3- Difficulty of using data and code repositories

One significant barrier that was mentioned by the vast majority of scientists is the difficulty of using data and code repositories; this difficulty emerged between computer science and non-computer science background scientists; they emphasised that these repositories are not straightforward and scientists need to learn how to deal with them. The Research Software Engineering group has been asked about their recommendations to make it easier for non-technical people to use such repositories. It was noticeable that their answers converged on two main solutions: training scientists to use these tools smoothly and improving the tools themselves to become more useable.

5.2.3.1- Training

Participant 1 emphasised that as a short-term solution, scientists should be trained to be able to use these tools. Furthermore, he insisted that to use this kind of tools efficiently and effectively, they need to improve their computational skills during these training sessions.

As a member of a scientific journal, participant 2 admitted this was understood at the journal and they started to teach scientists who do not have programming skills to be able to use tools, such as GitHub; he said:

"We thought about this in the journal. the problem is something like GitHub, not all people can use GitHub, it is quite difficult to use if you are not a programmer, it is quite complex piece of software, there are a lot of complicated terms, it is not a copy paste source code repository, you have to understand all these things, I think it is quite hard."

In addition, he asserted that it is the university's responsibility to provide these training courses and justified his view, saying:

"the Universities' role is to teach scientists how to use these tools to make their data analysis available because different universities here in the UK are graded based on their research that they work, and if they don't made their analysis open, they will get lower grading."

Despite the fact that participant 3 is a member of IT services, she admitted that sometimes when she uses these tools, she needs to refer to their instructions to ensure that she did not get something wrong. She added, as an IT services provider, that:

"We thought about providing online materials, very simple online courses to go through and we have talked about producing YouTube videos, training videos because actually seeing somebody going through the process is useful and then additionally we are considering developing easy frontend tools for indentifying existing tools that will do this for some people."

Participant 5 asserted that the training course should be designed to be suitable for researchers in various scientific fields, including those who do not have a technical background. One of the

items of feedback that she received in one of her workshops was that a biologist came to her and said this was the first time that someone has explained something in an understandable way. People usually forget that not everyone has a computing background, so when they start to use terminologies that are related to computing, these people becomes lost.

Participant 4 had a slightly different opinion regarding training. He believes that training can only be successful in certain cases; it depends on whether people have time to attend these sessions, as well as on whether people have time to provide this training and because of that, code publishing is dependent on these tools. It is much better to have easier graphical user interfaces.

5.2.3.2- Improving code publishing tools

In the longer term, participant 1 asserted these tools must be improved, and work has already started to make them easier to use with a minimum effort; he said:

"People are working on tools to make it easier to work with repositories without knowing all the commands, so you can use GitHub interface for example, or the web interface or GUI tools that you can put your files in."

He added that some researchers had gone further:

"What we need is for example if it was data, you know some DNA sequences and you want to add some annotation to them, automatically this stuff will happen for you in the background to put into a version control, I am not seeing many tools like that."

Regarding this issue and specifically regarding having comfortable graphical user interfaces, participant 2 had a different point of view. He thinks that these interfaces will lead to these tools losing some of their functionality and they will thus be restrictive. For that reason, he argued that there should be some compromise here; if scientists really want to use these tools, they need to make an effort. If they want full functionality, they have to learn how to use them, how to programme them and how to use the command line interface. To address this functionality issue, participant 3 suggested that these tools could be a combination of a

command line and a graphical user interface; thus, experts can always have the command line if they want to perform tasks that are more complicated. In addition to having better graphical user interfaces, which covers basic cases, participant 4 proposed another solution that may make these tools more comfortable; he suggested that:

" If you could upload your code, and for the next version you also upload your code again, the same as the first time, so you upload it as a zip file, and now the system itself could work out what has changed and mark the changes directly rather than relying on you to resolve this synchronization between the two things so you just keep uploading zip files and it goes oh you have added these files, and this file was changed in this place, that could make it easier."

Participant 5 emphasised that these tools must not be designed as exclusive tools; the interfaces should be built in a way that researchers from various scientific areas can contribute, even if they do not have a technical background.

5.2.4- Establishing central repository

This idea was raised by one of the scientists who took part in the first-stage interviews; he mentioned it as a technical problem that he faced and in his view, he believes that having something such as a central repository through which data and code could be managed internally will encourage him as a scientist. As well, he suspected that this would be the same for other scientists.

When this idea was proposed to the Research Software Engineering group, the responses varied. Four of the seven participants emphasised there is an urgent demand for an internal repository; three of them were IT services members.

Participant 1 views that the idea of having a central repository within the university is like a double-edged sword; it has some advantages and some disadvantages. From one side, because it will be free and supported locally, having an internal repository could be preferable among scientists. On the other side, such a repository will definitely need special management and

control, and this will be an additional burden on IT services; having an internal repository will be the exact same as using a commercial one, such as GitHub, or even worse. He explained:

"If you are using something in the public space, it might be easier to collaborate with people who are not in the same institution. If I am using a repository here and I want to give my friend in Southampton access, maybe it is not possible, if I am collaborating on the web, I can give it to them and say ok you can connect to my repository as well, this might be not allowed."

He added

"There are some other types of data systems that I don't know if it is some kind of data marts or data warehouses; that is slightly different and that will be quite useful and quite expensive to run."

Participant 2 holds the view that this will be a good idea and justified his answer by emphasising that in a big university, such as the University of Manchester, several scientists work in the same research area; thus, having this central repository will enable them to determine what work has been done in that specific area thus far. In addition, such repository will be useful to store datasets that cannot be published.

Participant 3 totally agreed with this idea and said

"There is always an advantage to having things locally, you can control its access but most people know about GitHub, and you know we can create and have corporate accounts with them, so everything is managed by the university of Manchester, and you can still have public and private repositories, so it is still possible to put sensitive information in there."

When she was asked about the central repository and collaboration issue, she said that it depends on how open the research project is and upon the researcher him or herself, whether he or she is looking for collaborators from outside his or her institution for his or her research or not. It also depends on the funding agency. Therefore, if the scientists want to publish their work as an open-source project, places like GitHub are appropriate to achieve this.

Participant 4 has agreed with Participant 1 on the fact that a central repository may impede external collaboration. He said:

"The biggest issue with this is usually collaboration, so if you've got two people working on the same thing, maybe one said I don't want to put in your university repository. I think it ends up with that problem where you put it in somewhere like GitHub or something like that, it means its open to everybody, anyone can use it, and no one can claim ownership over it."

In addition, he also agreed with the idea of "providing support rather than the service itself"; he insisted on having a big private account in a public repository, such as GitHub, for the university, so researchers and scientists within the university can use it. This would be adequate and would avoid the university having to invest in managing the infrastructure of a central one. He sees this idea as being successfully applicable in one case: if the university can host an internal repository. As well, this local repository will be compatible with an external public one, so anything submitted in the local repository will be automatically dragged on the public one.

For a different reason, participant 5 also had this idea about having something internal that is compatible with a public resource, such as GitHub. She justified:

"I think it would be good to have a resource that is in somewhat compatible with external resources so for the time when you are working on your publication or still working on a bit of research, you put it there, but then there is an easy step to move it somewhere more public to make it really open science."

According to participant 6, having a central repository is beneficial in the long term; an in-house repository could function as a library or a knowledge base for the university, so if people left, the university could preserve its right to access data as a funding body for that project.

Despite the fact that it is beneficial and can be successfully applicable, it is possible that this is more of a cultural issue rather than a technical one, because the main reason scientists and researchers want to have an institutional repository is that they are afraid of the idea that they are going to put their experimental details in public resources before they are published. Therefore, they need to be convinced that even if something like a central repository is not

available, they still can publish their own code and data in an external one, and all their rights can be preserved from being violated.

5.2.5- Adopting cloud computing and virtual machines to facilitate reproducibility

The Research Software Engineering group was asked about the idea of performing others' experiments within a virtual machine hosted by a cloud provider to facilitate the reproducibility of scientific contributions. The experimenter will save a snapshot of the virtual machine, make it publicly available and cite it in all appropriate papers. How this idea will work properly and what potential challenges may arise are discussed here.

Participant 1 started by stating that people are avoiding using virtual machines because they are a heavy weight option. Despite that, as a journal reviewer, he said that open research software, which requires building the entire experimental environment as a step of the review process, for reproducing these experiments by using snapshots of the virtual machines is much easier and more accurate.

He also emphasised that scientists who intend to use this approach in their research need to gain some software engineering skills; they need to have reasonable computational skills, some programming knowledge and command line and version control skills.

The big challenge in his view is how researchers in other scientific fields can exploit such approach. He said:

"What this university is trying to do and other universities are trying to do is having research software engineers who help researchers to make more help to do things. I think that is a really important function because if somebody doesn't have certain amount of education and understanding, it is really going to slow him down if you expect him to learn all these things, so they need help because they know everything about their field but they don't know anything about computers."

Participant 2 considers this approach an effective solution for the software dependencies issue; the software dependencies will be preinstalled in that virtual machine so people who want to

use this software, don't have to worry about such dependencies. He has stated that building these virtual machines is a time consuming task; it needs time to package it up and test it to see if it works properly. Furthermore, it needs to be documented very well because without such documentation, it will be like a black box, difficult to be used by others.

As an IT services member, participant 3 said that it is their responsibility to provide training materials to teach scientists and researchers how to use these tools and build their own virtual machines. She said

"I think they need training materials or introductory courses from us to show them how to use these things, there are a lot of tools out there, so research IT needs to define recommended list of packages, or a smaller set and then provide training materials, guidance and support for them, tell them what the various stages to follow, so run these following commands and that will create a VM and then you can make that available to colleagues."

For non-technical researchers, she emphasised that the effort should be between the researcher and the research IT team; researchers must make some effort to learn how to deal with these tools; at the same time, they can ask IT services for technical support. In terms of the types of training, to gain the maximum benefits, she said that training should be diversified among introductory courses, manuals, videos and even face-to-face conversations.

Participant 4 had an opinion here; he suspects that as this approach is a cloud-based one, this can make it unappealing to scientists. He justified that most researchers do not really care about where they run their experiments as long as it is a private place with a reasonable run and response time. Regarding a reasonable run and response time, cloud computing can guarantee these aspects by running on multiple machines simultaneously; on the other hand, security could be an issue with this technique.

Another issue was raised related to the place in which scientists will start their experiments. If they started the experiments in their systems, it is then a burden to build a virtual machine just to release the software, he claimed. Rather than starting the experiment in the system and

then moving it to the virtual machine, he instead suggested encouraging scientists to start the work in that virtual machine in the first place.

Participant 5 and 6 added a suggestion to create a built-in tool that has a repository role, through which people can store, access and maintain their virtual machine images, thus making them publically available to be shared by others.

It is worthy to note there was a consensus on virtual machines as an effective solution that may have a great impact on scientific reproducibility. Seven out of seven individuals from the Research Software Engineering group emphasised the importance of providing proper training to ensure that people have some understanding of the basic aspects: what a virtual machine is, what cloud computing is and how to launch a virtual machine on that cloud. Participant 6 said:

"Workshops will be good. Just to give people the initial ground, what the concepts are and introduce the control panel and what bits do what. We need to provide people the appropriate information, an idea of what key components are there."

In the literature, Ince *et al*. (2012) mentioned that the lack of tools that are used to package the experimental details in the research papers is one of the technical barriers that scientists might face and that might hamper scientific reproducibility. Participant 4 stated that these virtual machine images could be an effective solution through which this problem could be overcome. He said:

"About experiment reproducibility, there is another thing that people have done which are VM systems, you build into a VM and you just treat that as a single thing and run it."

5.2.6- Code licensing and attribution

According to a survey conducted in 2009, a group of 723 academic researchers in the machine learning field were asked for the reasons behind their decision to share or withhold their experimental details, such as code and data. For 34%, the reason was copyright. During these interviews, the Research Software Engineering group was asked about licensing code and attributions.

There is a debate about whether publically funded research should be publically available. In contrast, there is a group of people who view this research as their work, and they have the right to make money by commercialising their research. Participant 1 sees the licensing issue as a hot topic in the field. For researchers in the universities, he insists that:

"It is important to realize that when you are a researcher in a university, you don't own the intellectual property, the university owns it, and so if you want to publish, you should probably get the permission from the university."

He also emphasised that scientific work should be licensed in a sensible and compatible way. For example, some scientists have no problem publishing their data, but not the scripts, and the opposite is also true; the other group argues that they will publish their scripts, but not the data. He said that in scientific reproducibility, both data and scripts are needed to rerun the experiment.

Participant 3 stated that a part of the problem is in understanding the different licensing types, as sometimes they can be quite complex; researchers do not recognise what they should and should not use. Participant 5 also supported this claim by recounting a practical experience; she said:

"We did a bit of poking around and we asked people what are your institutions policies regarding intellectual property, licensing? Most people either don't know or saying what they thought they know. A lot of people don't know where to ask."

However, before increasing scientists' knowledge regarding these different types of licenses, participant 3 emphasised that a change should be made at the cultural level. Their conviction should be changed by trying to push things to be open source.

Participant 4 asserted that training and well-written documentation are keys to increasing the awareness of various licensing types and how they can be used in different situations.

5.2.7- Other suggestions to help scientists to make their scientific contributions reproducible

Finally, the Research Software Engineering group was asked about any additional recommendations as to how scientists could be supported to make their scientific work reproducible.

Participant 1 emphasised that it is the institution's responsibility to take this role and try to help researchers to achieve this aim. According to him, scientists can contact several expert bodies to afford such assistance. There are Data and Software Carpentry that provide a two-day course to help scientists and PhD students to improve their software and data handling skills. In addition, there is the SSI, which has the aim of increasing the lifetime of the software.

Participant 2 went further and said that scientists need to be convinced to change their publishing culture in the first place, and this could be accomplished by increasing their awareness about the importance of reproducibility to the scientific community. He also suggested that universities can encourage scientists to publish their code and data by measuring how much these datasets and code are used by other people.

Participant 3 added that

"I can't think it comes direct to the training but maybe something like support having community research software engineers on campus, and not just research software engineers in computer science and research IT but basically anybody is on campus that doing a significant work of software engineering. Moreover, creating research software engineering club, so people can turns up in a room, some people do presentations, and we can present this kind of information and discussion."

Participant 4 had a different opinion regarding training; he is convinced that encouraging people to attend these training sessions is not easy; just because you are offering training does not make them want to attend. "It's the usual training issues, the time VS the resources available and people that can do the training", he said.

To encourage scientists to practice openness in their publications, participant 5 suggested giving them a reward for the fact that they have learned how to adopt open science aspects in their publications, because the problem is that researchers get rewarded only for publishing a paper. Therefore, if anything is going to take their time away from publishing a paper, they ignore it.

Cambridge University already has experience in bringing scientific reproducibility standards into classrooms by providing an eight-week course in which students can learn reproducibility standards and then reanalyse some papers in their field as a practical application. The Research Software Engineering group has been asked about adopting this practice in other universities.

Seven persons out of seven agreed that this is a good idea, and it would be worthwhile to adopt in other universities within the UK. "I think it is useful if there is a lecture or practical based on scientific reproducibility", participant 2 said. Participant 5 supported this view by saying:

"I think bringing in concrete examples is useful, you know, let's take this and try to reproduce that and then realize oh that is not easy what I am going to face and this is what other people would face if I don't put my resources there in a reproducible manner."

Participant 6 suggested that these courses to be provided at the school level or faculty level as unregistered undergraduate courses. Computer science schools maybe good candidates for providing these courses, he said. Regarding teaching this type of course at the undergraduate level, participant 4 said:

"If it is the only part of research, it has to be more of an advanced stage in some ways, because people coming at undergraduate level may not go to do research, so it is not a part of what they do in everyday life, and their work. It has to be optional."

5.3- Conclusion

Within this set of interviews, the technical barriers that were identified by the scientists have been individually addressed with the Research Software Engineering group. They suggested some recommendations to overcome these barriers and facilitate the publishing process.

Regarding using the online code repositories, they suggested **developing a simple easy-to-use Graphical User Interface alongside the current command line interface**. **Tracking changes to be documented automatically in these tools** is another recommendation that was suggested by them. Moreover, they suggested **providing training to teach scientists and researchers how to use these repositories and training in some computational research aspects, such as data management planning; general software engineering skills and awareness about open source licensing systems** were suggested, as well. **Using specialist computational platforms to support reproducibility** has been also proposed as a useful solution to practice openness in publishing.

In the next stage, these solutions are evaluated by scientists via a questionnaire to determine how effective they are perceived to be and to examine whether such recommendations will encourage them to practice openness in their scientific publications.

Chapter 6

6- The survey results

6.1- introduction

As a result of the discussion with the Research Software Engineering group at the University of Manchester, several solutions to help scientists publish their code have been proposed. A set of recommendations were developed based on these solutions in order to overcome the technical issues that may prevent scientists from publishing their code. These recommendations were incorporated in a questionnaire aimed at scientists to determine how effective they are perceived to be. The sections below show the questions covering each recommendation in turn, followed by a description of the results.

6.2- The results analysis

The questionnaire was distributed to staff and PhD students within the University of Manchester and other universities within the UK including Leeds, Sheffield and Nottingham. Sixty-four responses were received.

Figure 6.1 Respondents' job titles

The first part of the questionnaire collected data about respondents' background. Researchers from various scientific research areas including computer science, information management, biology, ontology engineering, computer graphics, computer networking, computational biology, bioinformatics, computational physics, knowledge representation, scientific open data,

machine learning, health informatics, medical statistics, cognitive neuroscience, operations research, geophysics, biochemical simulators, infectious diseases, theoretical physics, nuclear physics, climate change, computational neuroscience, genomics and computer arithmetic took part in the study.

Discipline	Number of respondents	Discipline	Number of respondents
Computer vision	3	Geophysics	1
Research Software and Scientific Software	1	Computer science	5
Theoretical Physics	1	Computer Systems Research for many-cores	1
natural language processing	1	Computational biology	1
Computer Arithmetic, Neuromorphics, Formal Methods	1	Integration of spintronic devices to CMOS	1
Machine Learning	4	Computational physics	1
nanoscale devices and materials	1	information management	1
scientific open data	1	Data Management	1
Computational Neuroscience	1	Web Engineering	1
Bioinformatics	2	hardware/architecture/SoC	1
Sciences	1	Knowledge Representation	1
Cognitive neuroscience	1	medical statistics	1
particle & nuclear physics	1	health informatics	1
Materials modeling	1	data mining	1
Ontology Reasoning	1	computer graphics	1
Infectious diseases	1	Ontological Engineering	1
mathematics	1	Feature selection	1
Biology	2	Text mining	1
Genomics	3	eScience	1
online collaboration	1	Web, HCI, Accessibility	1
End-User Service Mashup	1	Oceanography / renewable energy	1
Formal Methods	1	Research software	1
operations research	1	Computer Networks	1
Climate Change	1	Professional software development	1
AI	2	Ontology	1

Table 6.1 Number of respondents for each discipline

Table 6.1 illustrates the number of respondents for each of the participating disciplines. Some answers were affected by the fact that 49 of 64 respondents belong to computer science fields. To some of the questions included embarks such as “I already use these tools in my daily work”, “I don’t need training in the above as I have had already” and “I’ve read a lot about the other topics already” reflected this fact.

As shown in Figure 6.1, the vast majority of the respondents were either PhD students or academics. The group included research associates, software engineers, industrial researchers, postdoctoral fellows, scientific programmers, librarians, institute directors, research assistants and senior research fellows.

6.2.1- Storing code in online repositories

Regarding storing code in online repositories, the difficulty of using these repositories sometimes deters scientists from using them as places to deposit code. One of the suggested solutions was to develop simple, easy-to-use graphical user interfaces (GUI) alongside the current command line interfaces.

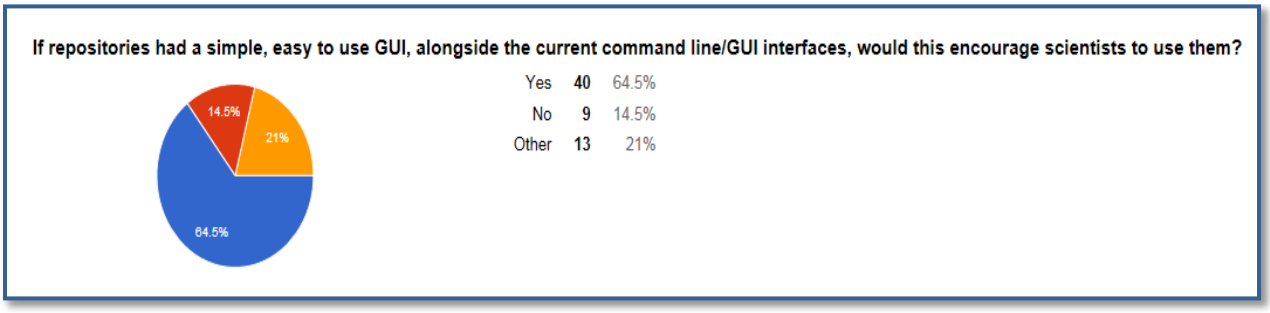

Figure 6.2 Repositories and GUI

Among the respondents as shown in figure 6.2, 64.5% agreed that GUI interfaces would be useful, while 21% of them had another perspective. Five said that they do not find command line interfaces difficult and they already use them in their work. It was noticeable that their answers were affected by the fact that they came from a technical fields thus using such tools is not an issue for them. One participant said that a GUI may “put him off” and another one said that “scientists who can code should be ok with using command line”.

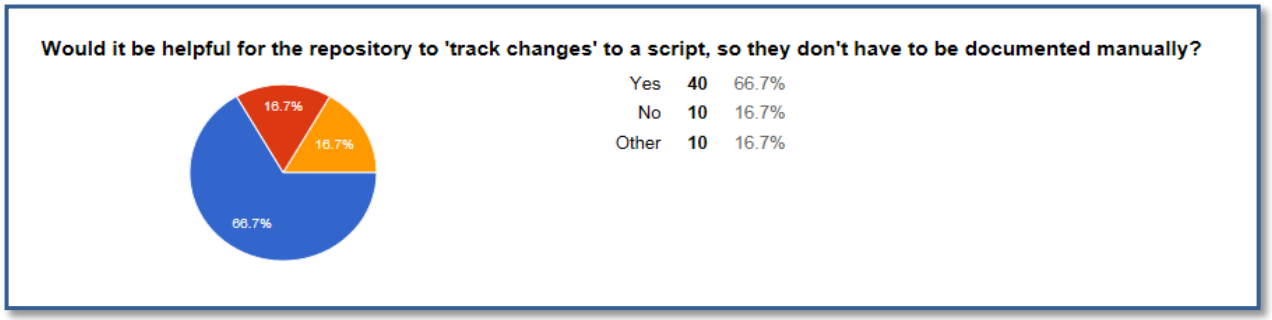

Figure 6.3 Tracking changes

With regard to tracking changes automatically rather than manually, figure 6.3 shows that 66.7% of the respondents approved of this idea. Two of the participants said that they did not understand the question properly, and two of them selected “don’t know”.

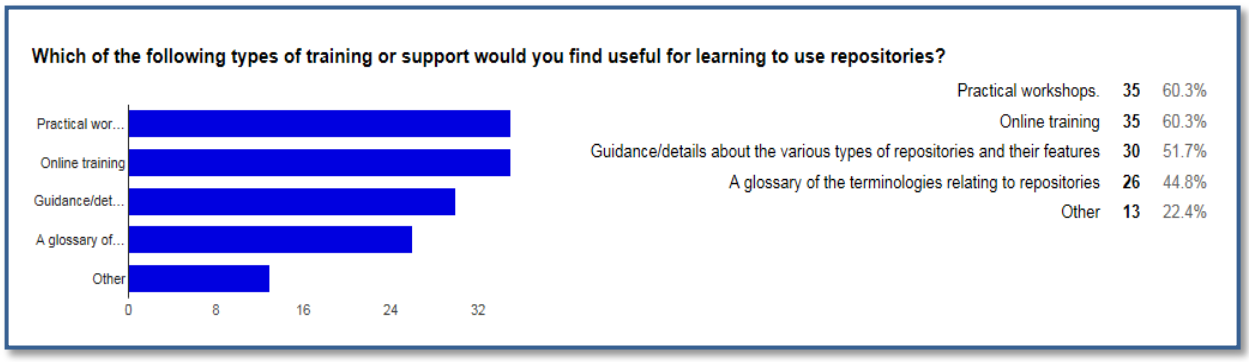

Figure 6.4 Training types

When the participants were asked about the type of training they needed to learn how to use these tools, 60.3% favoured practical workshops and online training (see figure 6.4). There was a convergence in percentages between other types. Some of the respondents indicated that they prefer tools that do not require any training. Two of them added that, in addition to workshops and online training, they need to have well-written documentation. One participant expressed the view that, rather than specific training, it is better to teach software engineering principles to students at the undergraduate level. Another participant indicated that advice from senior scientists in his field would be helpful, while another suggested that seminars about the practical issues could also help. Time was clearly an important factor since approximately half of the respondents preferred online training. This could also be because people can work through at their own pace and refer back to materials when necessary.

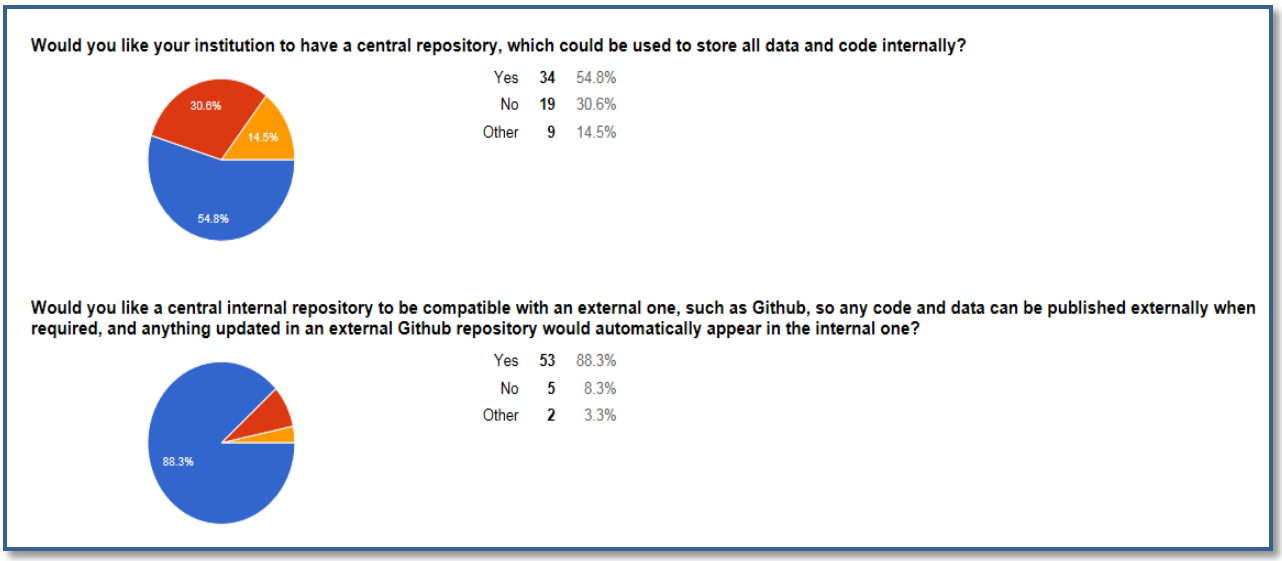

Figure 6.5 Central Repositories

Figure 6.5 shows two related suggestions. When the suggestion was to establish a central repository for their institutions, 54.8% of the participants agreed while 14.5% had different opinions; the most prominent alternative suggestion was “purchasing BitBucket accounts for students and academic members”. One participant justified his disagreement of the idea of central repositories by pointing out that this type of repository does not allow external collaboration, while another respondent emphasized that it should be easy to transfer to a public one when needed. When the suggestion was to make that central repository compatible with an external one, the percentage of agreement increased to 88.3%. One participant suggested having a public interface to the central internal repository as an effective solution. Another respondent suggested including the option to sync the two repositories automatically or manually.

One participant indicated that in the long-term, the cost of maintaining the central repository would be an issue that could lead to the university outsourcing the whole thing to GitHub, for example, and paying for a private repository; it is thus better to pay to have private repositories in an external one.

Another respondent, a research programmer, said that he works with faculties and their PhD students on a daily basis, and it is very surprising that they cannot use version control systems properly. He was wondering how to spread this practice. Moreover, he noticed that PhD students do use the repository if it is configured by someone else; command line tools are problematic for non-CS majors, he said.

Another participant indicated that these learning and training sessions should be provided at the early stages because, later, they would not have enough time to complete this training; he said that time is a real barrier along with the perceived difficulty.

Some of the participants expressed the opinion that the actual problem is cultural rather than technical, and the focus should be on that track. Whilst this may be true, interviews with scientists who already have experience in code publishing have mentioned that these technical issues are real barriers that hinder them in some cases.

6.2.2- Training in computational research

Figure 6.6 Training in computational research areas

Regarding the suggestion of having training in computational research aspects as shown in figure 6.6, 67.3% of respondents supported data management planning, and 44.2% wanted to increase their knowledge of intellectual property issues; 55.8% indicated a desire to learn more about open source licensing systems and general software engineering skills.

6.2.3- Scientific reproducibility

A number of institutions, including Cambridge University, hold workshops where students learn about reproducibility standards and then try to replicate the analysis of a published paper in their field, providing both theoretical and practical experience in reproducing research. The participants were asked whether they think it would beneficial to offer this type of experience at other universities. This suggestion was very popular, with 60 (95.2%) responding “yes”.

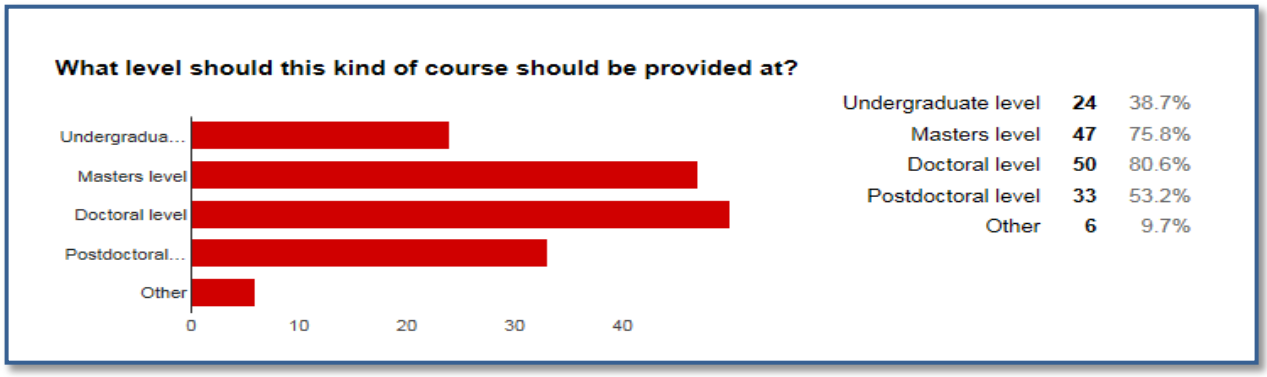

Figure 6.7 The appropriate level for the course

About the appropriate level for such courses, Figure 6.7 illustrates that the vast majority of participants think this kind of course is best suited to the postgraduate level. Two of the respondents expressed the opinion that including these types of courses in undergraduate or even master programs is pointless and complicated; they believe that working towards reproducibility is a skill that should be gained by PhD candidates before they start their PhD. Just 38.7% of them believe that it is appropriate for undergraduate students. It is worth noting that these answers are consistent with what the Research Software Engineer Group suggested regarding providing these courses at the postgraduate or PhD level.

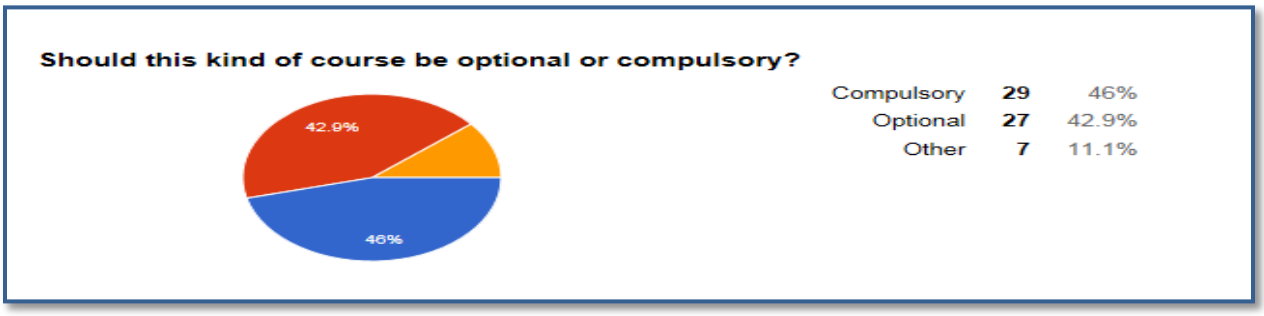

Figure 6.8 Course type

Regarding whether this course should be compulsory or optional, as shown in Figure 6.8, the respondents were split. One indicated that it should be optional because PhD students in theory-based domains would find it irrelevant, while another participant suggested that it should be a part of general training. Two respondents said that it should be compulsory at the PhD level, and another said that it should be compulsory for computational scientists. By contrast, one respondent expressed the opinion that making it compulsory would kill enthusiasm. Also, 31% of the respondents who recommended making the course compulsory were academics, which indicates that they are convinced of the importance of such courses for their students at the early stages of the research path.

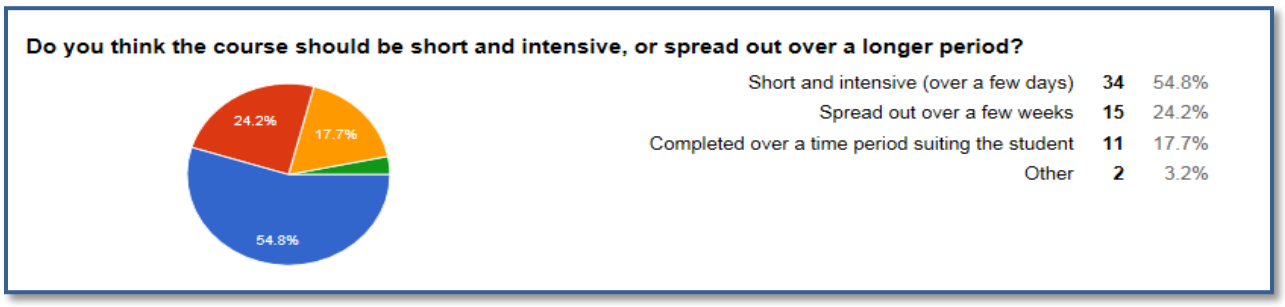

Figure 6.9 The appropriate period

When the respondents were asked about the appropriate period of time for this course, as shown in figure 6.9, 54.8% preferred a short and intensive course, which indicates that time pressure may have played a significant role in the participants' answers. One respondent said that if the course was spread out over a longer period of time, more could be covered, but this is probably less achievable due to tight availability of resources. Another indicated that this

type of course should be in the form of “ongoing support” to make scientific works fully reproducible.

6.2.4- Computing environments supporting reproducibility

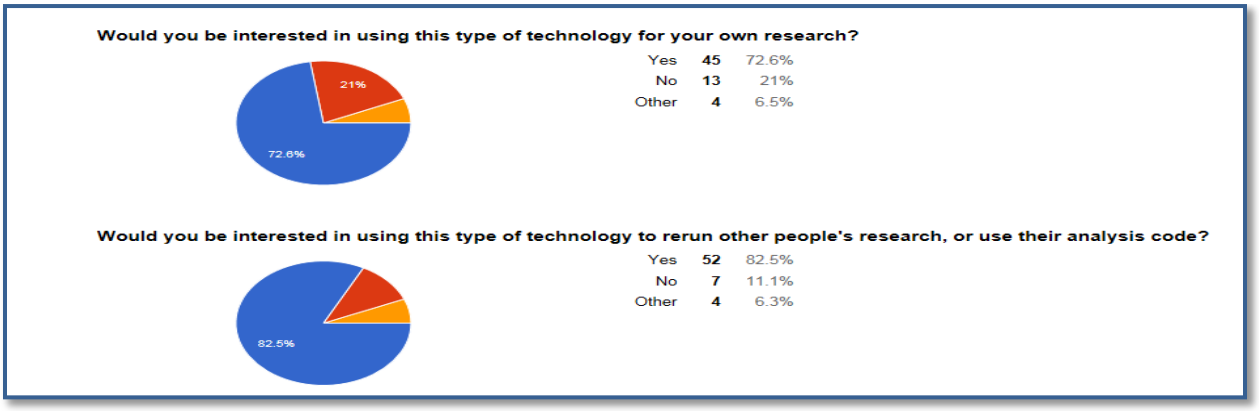

Figure 6.10 Using specialist platforms

The idea of using specialist platforms was proposed as a solution to facilitate reproducibility. These platforms allow a ‘snapshot’ of data and analysis software to be saved so it is easier for others to download and run the experiment.

The respondents were asked if they were interested in using such technology in their research. As Figure 6.10 illustrates, 72.6% were interested while 21% were not. One respondent said that his level of interest was dependent upon how difficult the tools would be to learn and use. When they were asked whether they would use these platforms to rerun other people’s research, or use their analysis code, the agreement percentage was significantly increased to 82.5%, which indicates that their decision in the first question may have been affected by other factors, such as the difficulty of using the tools.

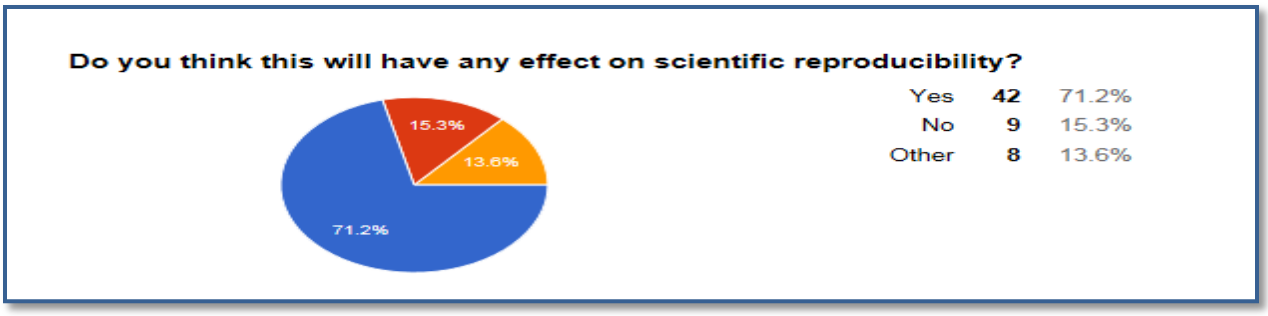

Figure 6.11 The effect of these platforms on reproducibility

When the respondents were asked if they think that these platforms will have any effects on reproducibility, as shown in Figure 6.11, 71.2% of the respondents indicated that these platforms have a positive effect on scientific reproducibility. Three respondents were not sure. One participant added “only if it is properly and commonly used” while another one said “yes if accompanied with changes in the way that papers are written”. Another respondent expressed the opinion that the biggest problem with such platforms, including Docker and other virtual machines, is the security issues that are raised during security updates. Two of the respondents said that instead of using complicated and inflexible platforms, they prefer popular tools such as GitHub to make their work reproducible. Another participant expressed the view that these platforms can help with reproducibility, but may undermine modification and extension.

It is worth mentioning that one of the responses was “it depends on how hard it is” which indicates that some of them did not realize how this approach works. Providing training and workshops on how such platforms work might increase the agreement percentage. The majority of respondents who selected “no” cited reasons such as security concerns, complexity and the belief that this cannot capture all the dependencies.

6.2.5- Other suggestions

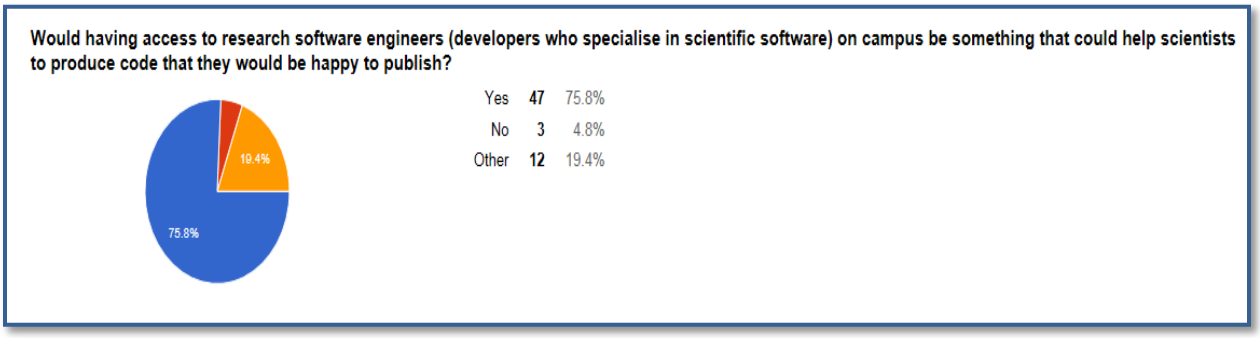

Figure 6.12 on campus software engineers

The idea of having on-campus support by research software engineers to produce high-quality code was viewed positively by 75.8% of the respondents (see Figure 6.12). 12 gave comments, but 3 said no. One said that this would be beneficial if it was involved in the earlier stages of the project. Another expressed the view that it depends on the researchers' programming skills; it might be beneficial for occasional users, who are not too confident in their coding skills. A third participant said that the problem is not producing the code but the process of maintaining that code. Another respondent said that "having a dedicated scientific computing person on campus would be like a dream come true". Consultations with software engineers may help with software quality to some extent, but require time investment, which is often the limiting factor, another respondent commented.

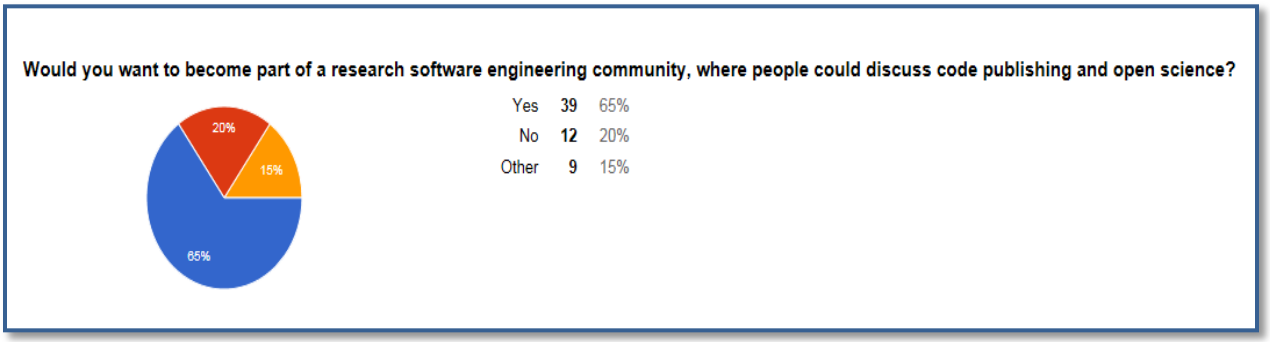

Figure 6.13 Software engineering community

One question asked the participants if they wanted to become a part of a research software engineering community, where people could discuss code publishing and open science. As illustrated in Figure 6.13, 20% answered "no". Five respondents indicated that time constraints would prevent them from joining such a community; one said that the community might affect their productivity as scientists. In addition, 7 of 12 people who were unenthusiastic about this suggestion were academics, which supports the hypothesis that the time factor played a role in their choice.

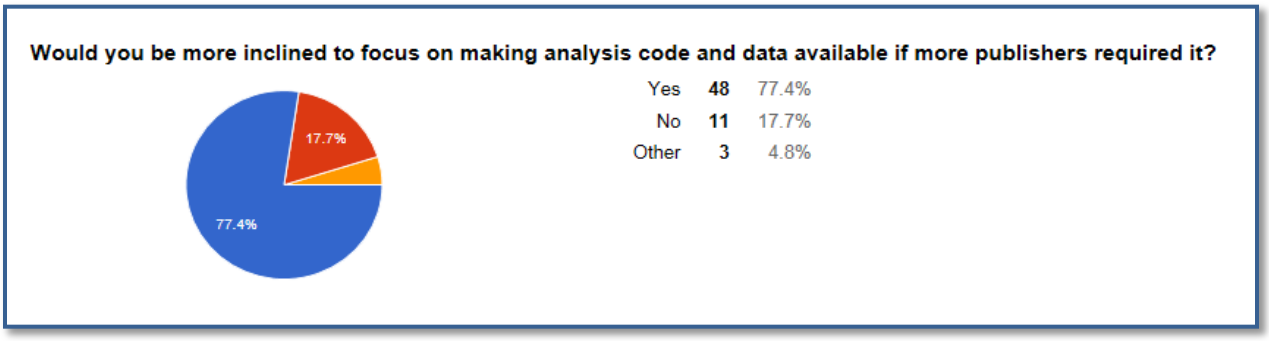

Figure 6.14 Requiring code publishing

In the last question, respondents were asked if they would be more inclined to focus on making analysis code and data available if more publishers required it. As shown in figure 6.14, the vast majority (77.4%) agreed that if the code was required, they would comply with that requirement, while 17.7% of them did not agree. One participant brought up the issue of

incentives. Since there are few incentives to encourage such practices, he suggested that the measures by which research is evaluated should be changed to achieve that aim. Another participant added that while there are clear benefits for publishing the software code, it should not be compulsory as there are circumstances (related to commercialization and interaction with industry) in which good science is done, but it is not advisable to release the software. If publishing high-quality papers required code to be made available, this level of coercion would have an effect, according to one respondent.

It is notable that the results of this study differ from what has been presented in the literature by Kattge et al. (2014) and the views expressed in the first-stage interviews. Kattge et al. (2014) and scientists who participated in the first-stage interviews were convinced that forcing researchers to submit their code would not necessarily make them publish it.

One respondent provided a list of his reasons for not publishing, including: some elements licensed commercially with exclusivity clauses; time to prepare the code and document it sufficiently for a general audience; the code may be evolving quickly and never quite get to a "finished" state and if core components were published, it makes it harder to make significant changes.

Respondents provided additional suggestions related to helping scientists publish code and data. One respondent said:

"If you decided to publish your code/data, try to make them stay available as long as possible. There are lots of cases that the early published source code/runnable application/data got cut off as soon as the related research is finished."

Another said:

"Learn the tools BEFORE you need them! Usually when the time comes for publishing your code, you are on a tight schedule, and too stressed to learn the required skills!"

One participant expressed the view that an effort should be made to ensure institutes' IP policies do not undermine the efforts of scientists to share their software and data publicly.

Finally, it is worth noting that the answers were skewed by the fact that the vast majority of the respondents (76.7%) have technical backgrounds. Moreover, it was noticeable from some of the answers that the respondents did not realize that the targeted audience included scientists from a variety of computational fields, not just technical fields.

Chapter 7

7- Recommendations

As a result of this project, the following set of recommendations have been put together to help scientists overcome the identified technical barriers and, thus, help them to publish their experiments' code. These recommendations have been welcomed by the SSI and will be published on their website:

- Improve the accessibility of code and data repositories:

As the difficulty of using code repositories is one of the barriers that prevents scientists from publishing their code, improving these tools would help them to overcome this issue. Developing an easy to use GUI alongside the command line can facilitate the interaction with these repositories for both technical and non technical scientists. 64.5% of the survey respondents found this solution helpful.

Another improvement that could be useful is to track changes in code versions automatically rather than requiring them to be documented manually, so the system itself can synchronize the changes in the different versions automatically. 66.7% of the respondents approved this idea.

- Train scientists how to use code and data repositories:

Another way to help scientists with using repositories is to train them on how to deal with these tools. This training could be provided through practical workshops, online materials and seminars about practical issues in relation to these repositories. To gain the maximum benefit, these training courses should be designed to be suitable for researchers in various scientific fields, including those who do not have a technical background. 60.3% of respondents were interested in learning these tools via online materials and 60.3% preferred practical workshops as their training tool.

In addition to training sessions, providing guidance about the various types of repositories and their features, a glossary of the terminologies relating to them and well-written documentation for them was supported by several respondents. 51.7% of the respondents stated that they need this guidance and 44.8% of them supported the idea of having a glossary.

- Establish an institutional repository:

The internal repository could be a private corporate account provided by an external supplier such as GitHub to avoid it having to be maintained internally. To maximize the efficiency and facilitate external collaboration, this repository should be compatible with an external one, such as GitHub, so any code and data can be published externally when required, and anything updated in the external GitHub repository would automatically appear in the internal one. An in-house repository could function as a library or a knowledge base for the institution, so scientists who work in the same research area can track what work has been done in that specific area thus far. Moreover, this repository will be useful to store datasets that cannot be published. 88.3% of respondents supported this idea.

- Provide on campus support:

Because scientists who do not have a technical background have a very limited grasp of programming, having access to research software engineers (developers who specialize in scientific software) will help them producing more usable, accessible and sharable code. This idea was viewed as helpful by 75.8% of the respondents.

- Encourage scientists to use computing environments that support reproducibility:

Scientists should be encouraged to use specialist platforms such as virtual machines that allow them to save a 'snapshot' of their data and analysis software to make it easier for others to download and run their experiment. These platforms facilitate the scientific reproducibility by enabling scientists to treat these images as a single object that is straightforward to run. In addition, this technology is considered to be an effective solution for the software dependencies issue.

This could be accomplished by providing proper training through workshops, introductory courses, manuals, videos and face-to-face conversations to ensure that scientists have a general understanding of the basic aspects of this technology. The suggestion of using this technology found helpful by 75.8% of respondents.

- Teach scientific reproducibility standards:

Provide a short, intensive course to teach postgraduate students the scientific reproducibility standards to gain both theoretical and practical experience in reproducing research. During this course, scientists should have the opportunity to replicate the analysis of a published paper in their field which enable them to gain several computational skills, understand the challenges involved in making research reproducible and produce legitimate publications. This suggestion was very popular, with 60 (95.2%) responding "yes".

- Improve scientists' computational skills:

This could be achieved by providing training and support in various computational research aspects such as data management planning, general software engineering skills, programming skills, awareness about intellectual prosperities issues and awareness about open source licensing systems. 67.3% of respondents supported data management planning; 44.2% of respondents wanted to increase their knowledge of intellectual property issues; and 55.8% of respondents indicated a desire to learn more about open source licensing systems and general software engineering skills.

- Establish a research software engineering community:

In this type of environment, people could discuss code publishing and open science issues and exchange their experiences regarding working in this area. 65% of the respondents found the idea of being a part of this sort of initiative interesting.

Chapter 8

8- Conclusion

8.1- Conclusion

The aim of this project was to develop evidence-based recommendations to motivate scientists to be transparent and to make their scientific contributions as accessible as they can.

The systematic literature review provided deeper insight regarding into scientists' attitudes regarding code publishing: what their views are, the reasons behind their decisions to withhold code, what kind of experiences they have gone through and what kind of obstacles they have faced.

By reviewing the literature, and after a set of interviews that have been conducted with a group of scientists who already have experience in code publishing to investigate what kind of barriers they have faced during publishing process, it was concluded that scientists could be categorised into two groups: those who are not convinced about code publishing for several reasons and those who are totally convinced of the importance of releasing the experimental code to the public but have faced some barriers that impede them from sometimes achieving their aim.

The first kind of barriers are cultural barriers and they relate to the scientists' conviction regarding releasing or withholding their experimental code. These barriers included the conviction that the code description is enough to understand what has been done, the source code is not expected to be released, as that the experiment's idea and results are clear and the overhead of preparing the code to be suitable for publishing. To address these issues, efforts should be made to change this culture and to motivate scientists to change their attitude to publishing.

The second type of barriers related to technical issues encountered when scientists intended to publish their code.

This project focused on these technical barriers and the solutions that were proposed by the Research Software Engineers group were evaluated by scientists through a questionnaire. This has resulted in a set of evidence-based recommendations that will be published by the SSI.

Although 76.7% of the survey participants have technical backgrounds, the significant level of support for the recommendations is evidence that scientists really need help; if even technical people are struggling, what about scientists from other non-technical fields?

8.2- Project limitations

During the work on this project, some limitations have—in some cases—affected its progress and results. These limitations were:

1- Finding a larger sample of scientists who are already publishing their code with whom to conduct the interviews. The interviews were conducted with eight scientists; seven of them have experience with publishing, but the last one was not convinced this was necessary.

2- Despite the attempt to distribute the questionnaire to other computation faculties within the University of Manchester and other universities across the UK to gain broader feedback, the results were affected by the computer scientists' opinions. Despite that the answers were skewed, actually this showed how important this issue is.

3- The time constraint has affected each stage of the research. Conducting the two stages of interviews, transcribing and analysing the data, designing and distributing the questionnaire, waiting for responses and analysing them—most of these tasks are time consuming and need a longer period to accomplish properly.

8.3- Future work

This project has addressed a set of technical barriers in detail; thus, evidence-based recommendations have been proposed and evaluated by a group of scientists through the questionnaire. Through reviewing the literature and the interview studies, it is worth noting that activating these recommendations is a non-trivial process and definitely requires effort to

be made at several levels. A cultural shift should be made to motivate scientists to make their work publically accessible, and various bodies, such as institutions and software engineers, need to be involved as well to achieve this aim. However, as not all these processes can be implemented in such a short period, a useful extension would be to apply some of these solutions to the available tools and gain the scientists' feedback on the new changes via a survey to discover whether these recommendations were found useful and practical to help scientists publish their code.

References

- Aronson, J., (1994). A Pragmatic View of Thematic Analysis. *The Qualitative Report,* 2(spring).
- Barnes, N., (2010). Publish your computer code: it is good enough. *Nature*, 467, p.753.
- Braun, V. and Clarke, V. (2006).'Using thematic analysis in psychology',*Qualitative Research in Psychology*, 3(2), pp. 77-101. ISSN 1478-0887
- Clarke, V. and Braun, V. (2013). Successful Qualitative Research: A Practical Guide forBeginners. SAGE Publications.
- Clarke, V. and Braun, V. (2014). Thematic analysis, in Teo, T. (ed.) *Encyclopedia of Critical Psychology*. New York: Springer, pp. 1947-1952.
- Donoho, D.L, (2010). "An invitation to reproducible computational research". *Biostatistics*, 11(3), pp. 385-88.
- Flick, U. (2009). *An Introduction to Qualitative Research*. 4th edn. SAGE Publications.
- Gill, P., Stewart, K., Treasure, E. and Chadwick, B. (2008). 'Methods of data collection in qualitative research: interviews and focus groups', *British Dental Journal*, 204, pp. 291-295.
- Grubb, A.M. and Easterbrook, S.M., (2011). On the lack of consensus over the meaning of openness: An empirical study. *PLoS ONE*, 6 (8), e23420.
- Guest, G., Namey, E., and Mitchell, M. (2012). *Collecting Qualitative Data: A Field Manual for Applied Research*. Thousand Oaks, CA: Sage Publications.
- Hancock, B., Windridge, K. and Ockleford, E. (2007). *An Introduction to Qualitative Research*. The NIHR RDS EM/YH.
- Hruschka, D., et al.(2004). 'Reliability in Coding Open-Ended Data: Lessons Learned from HIV Behavioral Research', Field Methods, 16(3) pp.307–331.
- Joffe, H. (2012). Thematic analysis, in Harper, D. and Thompson, A.R. (ed.) *Qualitative Research Methods in Mental Health and Psychotherapy: A Guide for Students and Practitioners*. Wiley-Blackwell, pp. 209-223.
- Kattge, J., Díaz, S. and Wirth, C., (2014). Of carrots and sticks. *Nature Geoscience*, 7(11), pp.778-9. Available at: http://dx.doi.org/10.1038/ngeo2280.

- Kelle, U., (1997). *Computer-assisted Analysis of Qualitative Data*. The LSE Methodology Institute.
- Kovac'evic, J., (2007). How to encourage and publish reproducible research, Depts. of Biomedical Engineering & Electrical and Computer Engineering Carnegie Mellon University. *Computer Engineering*, pp. 1273-76.
- Landis, J. and Koch, G. (1977). 'The measurement of observer agreement for categorical data', *Biometrics, 33*, pp. 159-174.
- Lenoir, M. et al., (1999). Intercepting moving objects during self-motion: effects of environmental changes. *Research quarterly for exercise and sport*, 70(7386), pp.349-60. Available at: http://dx.doi.org/10.1038/nature10836.
- LeVeque, R., 2012. Top Ten Reasons to *Not* Share Your Code (and why you should anyway). *Faculty.Washington.Edu*, pp. 1-6. Available at http://faculty.washington.edu/rjl/pubs/topten/topten.pdf.
- McClure, R.D. (2002). Common Data Collection Strategies Effective in Qualitative Studies Using Action Research in Technical/Operational Training Programs. Available at: http://evokedevelopment.com/uploads/blog/commonData.pdf, (Accessed: 28 July 2015).
- Meadows, K. (2003). 'So you want to do research: Questionnaire design', *Nursing Standard: Official Newspaper of the Royal College of Nursing*, 5(12), pp. 53–54.
- Nkwi, P., Nyamongo, I. and Ryan, G. (2001). *Field research into socio-cultural issues: Methodological guidelines*. Yaounde, Cameroon, Africa: International Center for Applied Social Sciences, Research, and Training/UNFPA.
- Northway, R. (2002). 'Commentary',*Nurse Researcher*, 10, pp. 4-7.
- Peng, R.D., (2012). Reproducible Research in Computational Science. *Science*, 334(6060), pp. 1226–27.
- Ritchie, J. and Lewis, J. (2003). *Qualitative Research Practice: A Guide for Social Science Students and Researchers*. SAGE Publications.
- Software sustainability institute. [Online] Available from: http://www.software.ac.uk/ [Accessed: 20th April 2015]

- Stanford University, (2011), Using NVivo for Qualitative Data Analysis. Available at: http://web.stanford.edu/group/ssds/cgi-bin/drupal/files/Guides/UsingNVivo9_0.pdf
- Stemler, S. (2001). 'An overview of content analysis', *Practical Assessment, Research & Evaluation*, 7(17).
- Stodden, V, (2010). Reproducible Research: Addressing the Need for Data and Code Sharing in Computational Science, *IEEE,* pp. 8-13.
- Stodden, V., Hurlin, C. and Pérignon, C., (2012). RunMyCode.org: A novel dissemination and collaboration platform for executing published computational results. *2012 IEEE 8th International Conference on E-Science, e-Science 2012*.
- Thimbleby, H., (2003). Explaining code for publication. *Software - Practice and Experience*, 33, pp. 975–1001.
- Vaughan, R.T., (2009). Publishing Identifiable Experiment Code And Configuration Is Important, Good and Easy. pp. 1–11.
- Woods, P. (2006).*Qualitative Research*. Available at:http://www.edu.plymouth.ac.uk/resined/qualitative%20methods%202/qualrshm.htm,(Accessed: 28 July 2015).

Appendix A

Stage one interviews questions

- What research area that you are interesting in?
- How many papers you publish before?
- Were any of them include any details such as code, data, and algorithms?
- Have you asked yourself if you did enough to allow others to verify and reproduce your findings?
- What is your general understanding about reproducibility and do you think that it is important in scientific research?
- Did you ever try to publish a paper in a scientific journal and they asked you to provide your code?
- If that happened, did it motivate you to clean up your code next time?
- Did you ever try to perform an extended research and the absence of the code was your only obstacle?
- Have you ever been in a situation that someone contacted you asking for your code? What your response was?
- If you knew that there are some institutions which offer some services to help you to enhance your code, does something like this will encourage you to share your code?
- In terms of citations, do you think that there is a direct relation between code publishing and citation rates? Do you have experience regarding this?
- Do you think that journals practice enough efforts regarding encouraging scientists to release their code?
- In your opinion, do you think that publishing code have some benefits? What are these benefits?
- Finally, do you have any recommendations that may help other scientists to start practicing openness in their publications?

Appendix B

Stage two interviews questions

1- One of the technical barriers that were mentioned by the respondents is that they don't have places to put their staff in, there is no sensible sources to put their experiments details in. **how something like this could be addressed in your opinion?**

2- Another barrier which was mentioned by more than a participant and mentioned in the literature as well is the lack of tools that used to package the experiments' details in a single file, "the data are scattered among different islands" a respondent said. Linking all these bits together is the key here; they want to find everything in one place. **How to make it easier to link or maintain links between different bits, and what the appropriate sources of things?**

3- One of the significant barriers is the difficulty of using data and code repositories; such difficulty was for both CS and non CS background scientists. **What do you think, how we can address this issue? What do you suggest to make it easier for non technical people who is not their every day job to use the? Is it useful to train them or show them how these staffs could be used?**

4- What about establishing a special repository for this university which will enable scientists to publish all their details in it. Should the university make it easier for them by providing one?

If yes, how we can resolve the conflict and the tension of having something local VS global? The conflict of make it easier for people in university of Manchester to do staffs and having a system for sharing and getting the things away from the wider to find it and that maybe a reason for some people to not use central repositories?

5- Bill Howe has proposed a solution to facilitate reproducibility in scientific contributions. The idea behind this solution is that performing others' experiments within a virtual machine hosted by some cloud provider. When the experiments are complete, the experimenter will

save a snapshot of the virtual machine, make it publicly available, and cite it in all appropriate papers. Readers of the paper who wish to reproduce the results can launch their own instance of the author's virtual machine, so the author does not need to re-package their code for multiple platforms, and the reader need not install additional software or debug portability problems. **How do you think it could work best, if you advice scientists to do this, what do you think to be required? How could we train people to use this, what the potential challenges that may rise here?**

6- How do you think we should deal with licensing code and attribution?

Victoria stodden has proposed a new licensing methodology to encourage scientists to publish the entire experiments' details, all the components required for reproducibility. By using this methodology, the licensing will applied to every aspect of the research, the research paper, the experiment code, the results of that experiment and any related material. It means that any paper published using one of these components will directly attribute the original author.

- How this can fix current licensing issues?
- How this might actually work?
- what will happened if the scientist used different data and code sources in which different licensing systems were used in his research, how to deal with this issue?
- How enforce people to respect this?
- How this methodology conflict with SW licensing, with derivative work?
- Why we don't just use something already there like creative comments?

Finally, do you have any other suggestions, how do we could support or train scientists better to help them making their work reproducible?

Appendix C

Stage one interviews sample description

Participant number	Gender	Research area
P1	Male	HCI, web interaction
P2	Male	Visualization of image data
P3	Male	HCI
P4	Male	Text mining
P5	Male	Modular reasoning
P6	Male	Data mining
P7	Male	HW SW interaction

Stage two interviews sample description

Participant number	Gender	Working area
P1	Male	Member in myGrid team and SSI
P2	Male	Member in gigascience journal
P3	Female	Research IT
P4	Male	Research IT
P5	Female	Member in myGrid Team and SSI
P6	Male	IT Services
P7	Female	Research IT