EUNIS 2019: The CodeRefinery Project

Bjørn Lindi[1], Radovan Bast[2], Thor Wikfeldt[3]

[1]NTNU, 7491 Trondheim, bjorn.lindi@ntnu.no
[2]UiT/The Arctic University of Norway, 9037 Tromsø, radovan.bast@uit.no
[3]KTH,Kungliga Tekniska Högskolan, SE-100 44 Stockholm, kthw@kth.se

Keywords
CodeRefinery, NeIC, FAIR, Open Science, software management, git, GitHub

1. ABSTRACT

CodeRefinery is a project established by the Nordic e-Infrastructure Collaboration (NeIC) with the ambition of teaching researchers Git(a version control system), collaborative tools like GitHub, and good practices for dealing with source code and data. Large focus is placed on reproducibility, maintainability and sustainability in developing scientific software. The project has given more than 20 workshops across the Nordic region over a two-and-a-half-year period with more than a total of 500 participants. The project members have found that there is a strong need for software engineering skills across a wide range of scientific disciplines. This paper discusses experiences of the CodeRefinery project.

2. INTRODUCTION

As CodeRefinery was launched by NeIC in late 2016, the goal was to provide Nordic Research groups and projects with training material and activities, technical expertise, and software development e-infrastructure in order to address the growing needs of modern research communities, especially helping groups to better deal with software complexity. Entering the project's second phase, the goal has now transformed into advancing FAIRness (Findable, Accessible, Interoperable, Re-useable) of research software, so that research groups becomes better suited at handling their code base according to good software management principles.

CodeRefinery argues that FAIR principles applied to research software should be a requirement for Open Science, but it is rarely explicitly mentioned in Open Science contexts. The OECD defines Open Science as "to make the primary outputs of publicly funded research results - publications and the research data – publicly accessible in a digital format with no or minimal restriction." Clearly, the research software is not mentioned, but reproducing or validating conclusions from contemporary open research data will require the software used in the first place.

According to Chen et. al., open data and open publications are not enough. "It (data) needs to be accompanied by software, workflows and explanations." Research software, including the context where it is used must also be accommodated by FAIR principles. As Chen et. al further argues, "having the reuse of research results as a goal requires the adoption of new research practices during the data analysis process."

This is similar to what CodeRefinery experiences. Many researchers and scientists have realized that they need new skills to prepare for the change to more reuseable and collaborative research processes. At the same time many scientific disciplines are becoming more computationally driven. Accordingly, managing software becomes a necessity. How this efficiently can be done and combined with the discipline specific research process is not necessarily obvious. As a result, a need for proper training has emerged across many disciplines.

In this context, the CodeRefinery project aims to provide unbiased training in best practices in software management. This is very much welcomed and appreciated. The feedback from the

workshops show that the participants truly have gained new skills, and that they are better suited to make use of Open Source Software, and work collaboratively with software and data.

3. OPEN SCIENCE AND REUSABLE RESEARCH SOFTWARE

The Open Science and Research Initiative, 2014 show that Open Science involves a systematic change to the way science and research is done. The principle of openness is extended to the whole research cycle, as shown below with the figure from the Foster Portal (Figure 1).

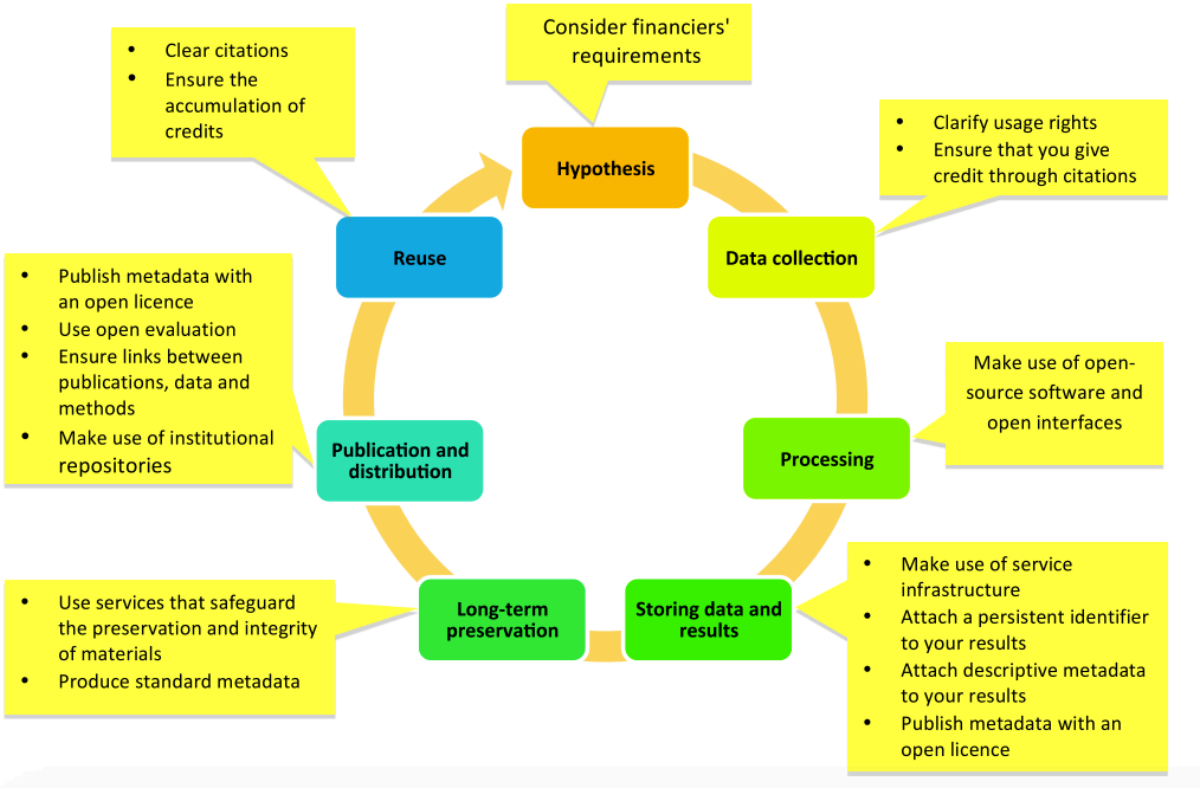


Figure 1

A closer look at “processing” (Figure 2) reveals that to “make use of open source software and open interfaces” involves several sub-processes requiring individual soft skills, as well as policies governing collaboration and community behavior. This comes in addition to the discipline specific research process.

At the individual level there are the skills with regards to working with software, including connecting software versions with published data and publications. It can be combining the discipline specific research process with software engineering, or making results findable and accessible with the use of Digital Object Identifiers and releasing versions. Proper test mechanisms and documentation must be made for the software and data to be inter-operable and re-useable. This is a must for proper verifications.

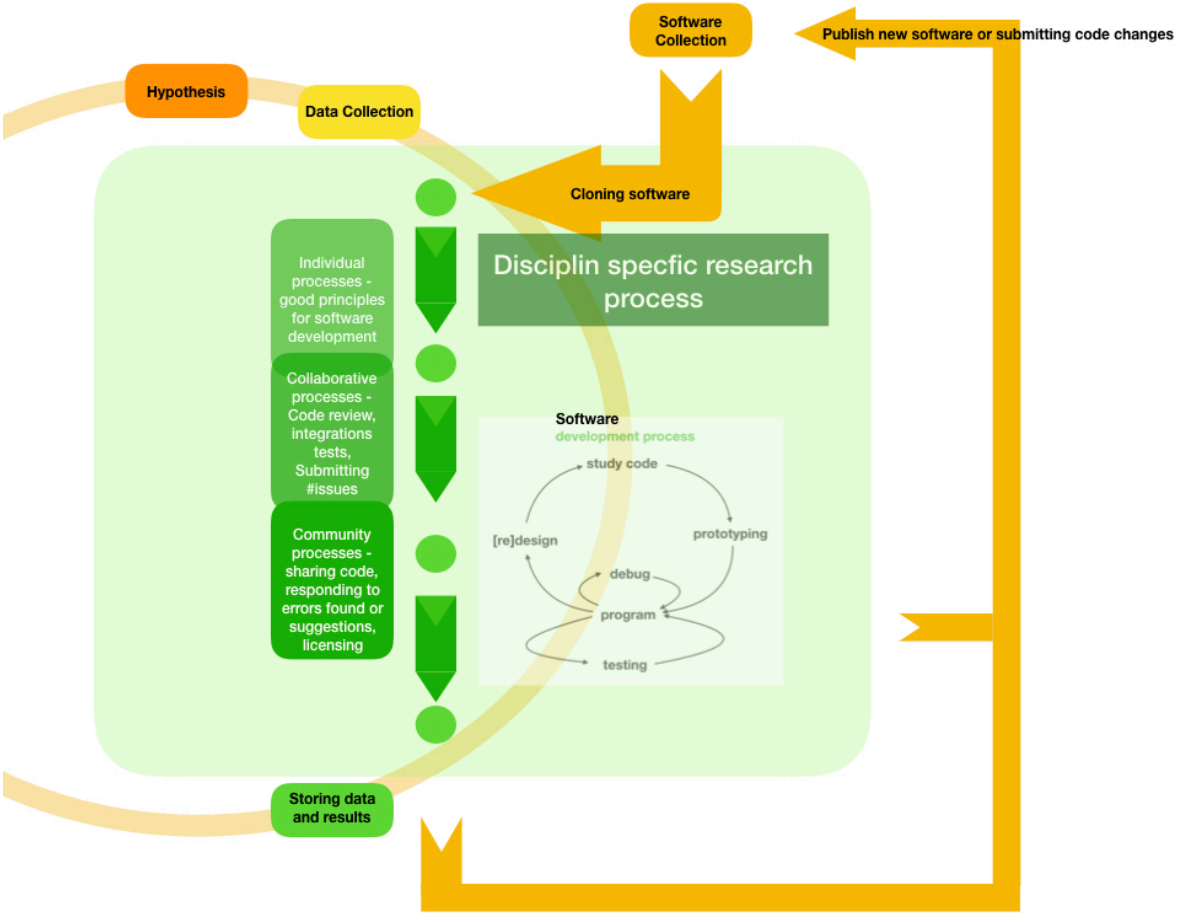


Figure 2

At the group or peer-to-peer level, there is the collaborative processes, like code-review, discussions involving feature development, and prioritizing which errors and issues to handle. These discussions can contain a lot of discipline specific value as they illuminate how research software got to its current state. The discussions may have broad community interest.

A third level is the community part, which can involve proper handling of software errors as well as requests for new software features. Very large Open Source code bases even establish User Forums as a community effort.

While software lifecycle can seem intrinsic to the current Open Science ambitions, it also has a life of its own. It parallels the data lifecycle, and it has the potential of becoming very resource demanding. Nonetheless, the first to benefit from established software management practices and a focus on reproducibility is often a research group itself. Markowetz makes this point in his paper and speech “Five selfish reasons to work reproducibly”.

The High Energy Physics (HEP) community is very explicit on re-use and proper software management as future progress in their field very much depends upon ability to reuse best-practices, tools and open source software from industry as well as from academia. Albrecht, J., et al. state this in “A Roadmap for HEP Software and Computing R&Ds for 2020s”.

The HEP Community states further very explicitly that training must be given more emphasis. Although other disciplines have not expressed such a coherent view, the same demand is present among individual researchers of different disciplines. This is something CodeRefinery team members experience in our encounters with workshops participants.

4. CODEREFINERY PARTICIPANTS SHOW THERE IS A DEMAND FOR TRAINING

The CodeRefinery Project has made its goal to advance FAIRness of research software, such that research groups can collaboratively develop, review, discuss, test and share and reuse codes.

Since the project was launched in late 2016, CodeRefinery has organized more than 20 workshops, with a total of more than 500 participants. The workshops have taken place all over Nordic Region including Estonia. Before attending the workshop, most participants answer a pre-workshop survey. Here is what the survey tells about the participants.

4.1. The participants and their background

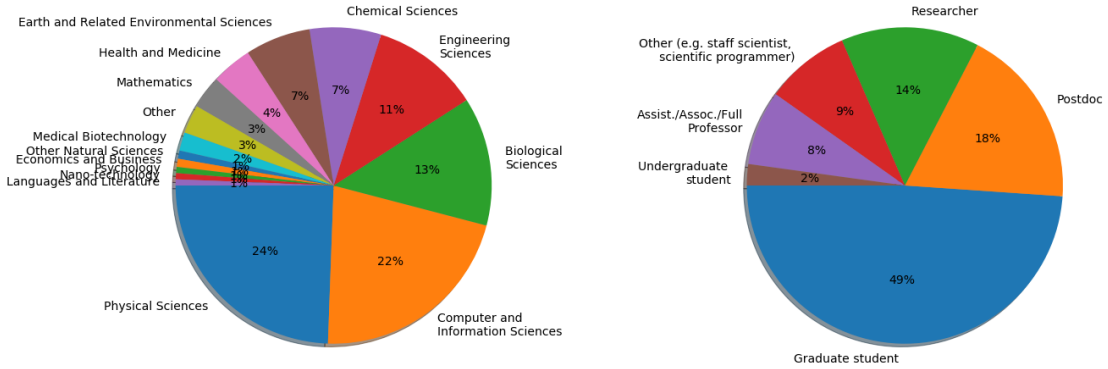
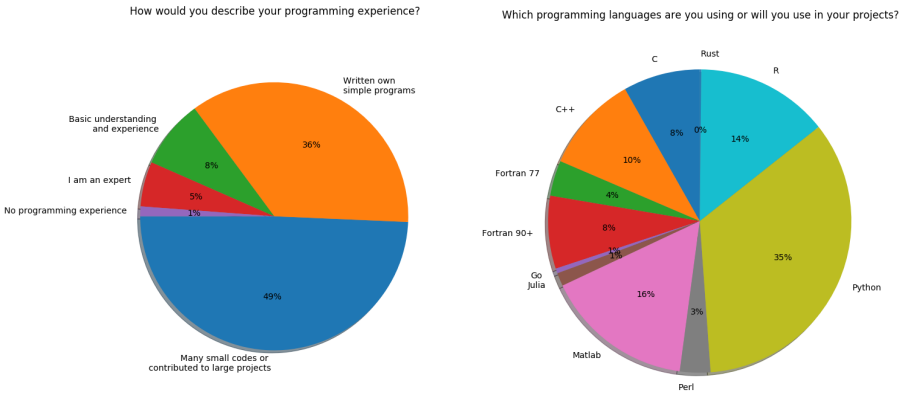
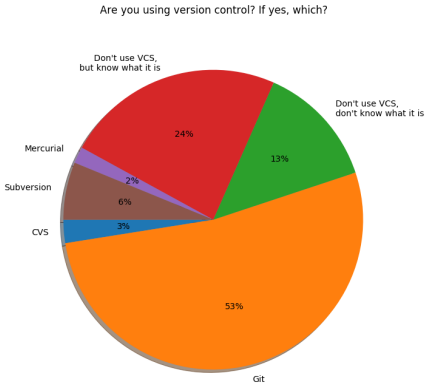


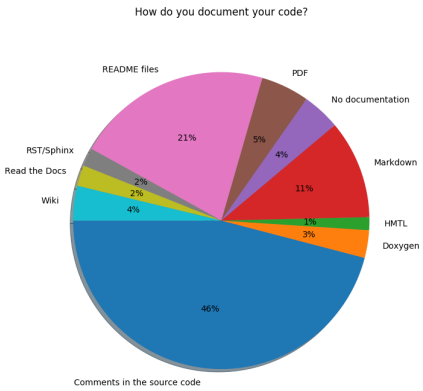
Figure 3

The participants' backgrounds are varied. They come from a wide range of disciplines. Physical Sciences are the largest group, followed by Computer Science. This is a somewhat surprising, since one would expect that what the workshop covers would be part of a Computer Science curriculum. Half of the participants are students, the other half are more senior.

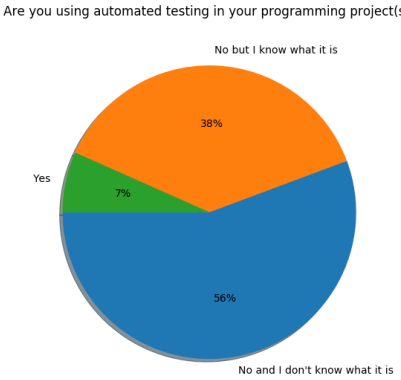


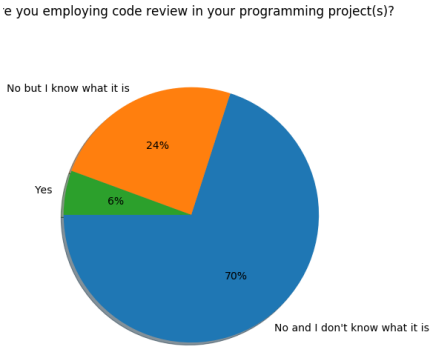
The participants' programming experiences differ, but very few have no experience. The largest group has knowledge of the Python programming language, but there is experience with many other programming languages as well.





A third of the participants do not use any version control system. The majority use comments in the code for documentation, followed by README files. Very few use public pages like “Read the Docs”.





A requirement for reproducibility is the verification of software programs – that they produce the expected results. Automatic testing through continuous integrations services like Travis, offer a good and efficient way of verifying software. Very few participants have experience with these types of services before the workshop. Code review is another way of improving code quality, and can be viewed as the software engineering’s counterpart to scientific peer-review. Nonetheless, very few have experience with this as well.

5. CODEREFINERY’S CONTENT AND IMPACT

A typical CodeRefinery workshop run over three full days. Workshop participants are invited to type along as CodeRefinery instructors go through these lessons. The format is copied from The Software Carpentries, with focus on hands-on tutorial interleaved with practical exercises. Participants have the means to stop the lesson if they loose the thread. Many participants share their own experiences, and there are often fruitful discussions during the lessons.

5.1. CodeRefinery workshop content

Most workshop follows the same format, but adjustments are made if the workshop pre-survey indicates that this can be necessary. This can be to increase or decrease emphasis on Python, depending upon the participants experience with the language, or talk less about compiled languages if there is fewer with experience with compilation.

The lessons offered during a workshop are:

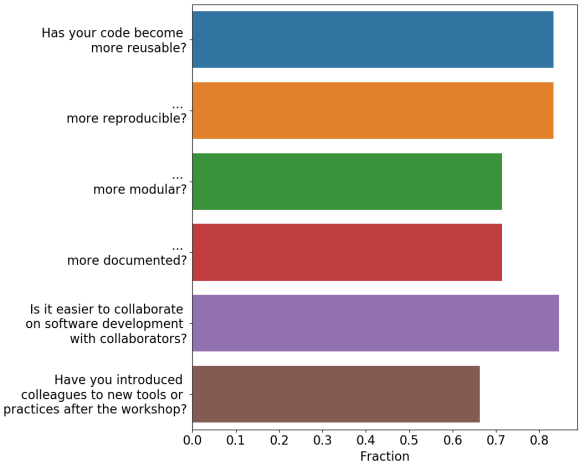
- Basic and collaborative git
- Git branch design
- Code documentation
- Automated testing

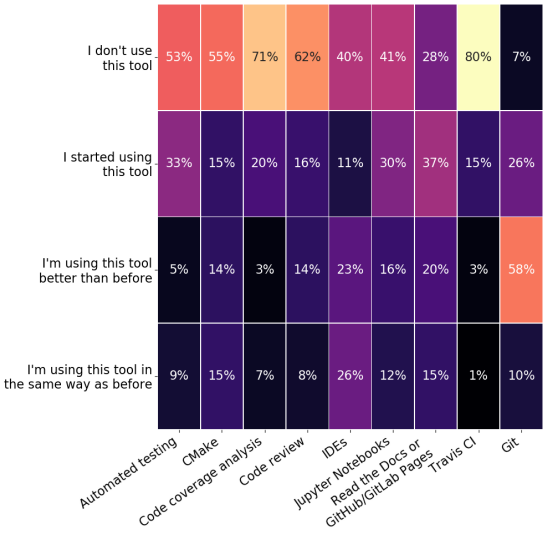
- Jupyter Notebooks/Jupyter Lab
- Integrated Development Environments
- Building portable code with CMake
- Social coding and open software
- Modular Code Development
- Reproducible Research

5.2. What is CodeRefinery's impact?

3-6 months after a workshop, CodeRefinery sends the participants a post-workshop survey. Here are the aggregated results of 92 respondents.

Most participants conclude that their code has better reproducibility and re-usability. It has become easier to collaborate on software development. Many have also introduced new tools to their colleagues.





More than half of the respondents are using git in a better way than before. Many of the new tools and techniques introduced during the workshop have a certain uptake, like automated testing, Read the Docs and Jupyter Notebooks.

Many participants report that their work process has changed. Quoting one workshop participant:

"The main thing I got out of the workshop is that I'm now extensively using the issue-tracking systems on GitHub and GitLab. Also on one of my major projects we have moved towards having shorter-lived feature branches and using merge requests for more frequent merges and less code divergence. I also started using GitLab's issues system more frequently. It made it easier for people in my lab to report bugs, and easier for me to keep track of them"

6. CONCLUSIONS

CodeRefinery is providing a service welcomed by researchers in the Nordic region, helping out to some extent with software and data challenges many researchers face. Currently the project is getting interest from institutions and interest groups outside the Nordic region. We firmly believe that it is necessary to promote software engineering skills throughout different research communities. The research communities need access to proper training in these matters. For Open Science (data, software and publications) to become a part of the on-going scientific production this is a necessary step.

The FAIR-principles must be applied to the research software, not only data. For research software to be FAIR it must be under version control with an accessible version history. It must be licensed such that it is reusable. It must have a digital object identifier (DOI). To ensure quality it should be reviewed. The environment and dependencies necessary for executing the software must be documented, preferably such that it can be re-instantiated.

CodeRefinery as a NeIC-project has a finite horizon, though it is the project ambition to establish a network of training resources promoting FAIRness in the Nordic research community. The project is actively working towards getting local groups all over the Nordics to become as knowledge resources and training providers.

Consequently, all CodeRefinery instructional material is made available under the Creative Commons Attribution-ShareAlike license. Anyone can build on CodeRefinery's instructional material or make use of it for their own purpose. CodeRefinery members hope that this can contribute to the establishment of a network of training resources, advancing FAIR-principles for research data and software.

7. COPYRIGHT NOTICE

8. REFERENCES

Albrecht, J., et al. (2018) A Roadmap for HEP Software and Computing R&Ds for 2020s. *arXiv:1712.06982v5*

Chen, X., Dallmeier-Tiessen, S., Dasler, R., Feger, S., Fokianos, P., Gonzalez, J. B., Hirvonsalo, H., Kousidis, D., Lavasa, A., Mele, S., Rodriguez, D. R., Šimko, T., Smith, T., Trisovic, A., Trzcinska, A., Tsanaktsidis, I., Zimmermann, M., Cranmer, K., Heinrich, L., Watts, G., Hildreth, M., Lloret Inglesias, L., Lassilia-Perini, K., Neubert, S., (2019). Open is not enough. *Nature Physics.*

Goodman, A., Pepe, A., Blocker A. W., Borgman, C. L., Cranmer, K., Crosas, M., Stefano, R. D., eta al. (2014). Ten Simple Rules for the Care and Feeding of Scientific Data. *Edited by Bourne, P. E. PLoS Comput Biol 10(4)(April 24)*: e100354.doi:10.1371/journal.pcbi.100352

Markowetz, F. (2015). Five selfish reasons to work reproducibly. *Genome Biology* 2015 16:274

Pasquier T., Lau M. K., Trisovic A., Boose E. R., Couturier B., Crosas M. (). If these data could talk. *Scientific Data 4, Article Number:170114*

CodeRefinery web page (2019). https://coderefinery.org

Creative Commons Attribution ShareAlike (2019). https://creativecommons.org/licenses/by-sa/4.0/

The Foster Portal (2019). https://www.fosteropenscience.eu/content/what-open-science-introduction

Nature.com (2016). https://www.nature.com/news/1-500-scientists-lift-the-lid-on-reproducibility-1.19970

The Nordic e-Infrastructure Collaboration. https://neic.no

OpenFoam User Community. https://www.esi-group.com/company/events/2019/7th-openfoam-conference-2019

Social Science's lessons for business (2019). https://www.strategy-business.com/blog/Social-sciences-lessons-for-business

The Software Carpentry (2019). https://software-carpentry.org

9. AUTHORS' BIOGRAPHIES

Bjørn Lindi works as a senior engineer at the HPC-group, NTNU IT-division. He has been working in this group since 2004. He has MSc in Electrical Engineering from NTH.

Radovan Bast works at the High Performance Computing Group at UiT - The Arctic University of Norway in Tromsø and leads the CodeRefinery project. He has a PhD in theoretical chemistry and contributes to a number of quantum chemistry programs as a code developer.

Thor Wikfeldt works as Application expert at the PDC Center for HPC at KTH. He has a PhD in chemical physics. He is the training coordinator at PDC and provides advanced user support for HPC users