Crowd-Sourced Exemplars for Developing Research Software Skills in STEM

James DesLauriers, Chris Cooling, John Pinney, Liam Gao, Katerina Michalickova and Deslauriers, J.

A paper presented at the 17th International Conference on Education and New Learning Technologies, 30 June-2 July, 2025, Palma, Spain.

The published version is available online at:

https://doi.org/10.21125/edulearn.2025.2184

CROWD-SOURCED EXEMPLARS FOR DEVELOPING RESEARCH SOFTWARE SKILLS IN STEM

J. DesLauriers, C. Cooling, J. Pinney, L. Gao, K. Michalickova

Imperial College London (UNITED KINGDOM)

Abstract

As scientific research becomes more data-intensive, STEM students supporting research projects or working on a master's project or doctoral thesis have had to learn to approach and solve problems computationally. These students come from varied backgrounds, and few have a formal education in computer science or software engineering. As a consequence, the area of *research software training* is becoming an increasingly important topic in academia.

ReCoDE is a research software training resource at Imperial College London, which proposes a collection of code exemplars that are sourced from the PhD community at Imperial. The project is led by the Research Computing and Data Science team, situated in the Early Career Researcher Institute and is supported by the central Research Software Engineering team. Each exemplar demonstrates a best practice computational approach to a particular research problem, which learners can engage with independently to further their knowledge on a particular programming language, specific software, or other digital skill.

ReCoDE involves several steps, including the selection, development, publication and evaluation of exemplars. This process embeds best practices in software project management and uses open-source technologies exclusively. While still a nascent resource undergoing refinement, we imagine that ReCoDE may one day inform efforts to curate user-generated content in other educational settings.

Keywords: Research software, programming, education, training.

1 INTRODUCTION

More and more, scientists and researchers rely on computational skills to approach and solve research problems [1]. Academia has long recognised the need to produce better research software, with organisations like The Carpentries [2] and the Software Sustainability Institute [3] leading the way in teaching best practices in software and data.

The Research Computing and Data Science[1] (RCDS) team, situated in the Early Career Researcher Institute[2] (ECRI) at Imperial College London, delivers short courses for postgraduate students, postdoctoral researchers, and research fellows. Our portfolio of courses[3] is varied, ranging from programming languages to the use of specific data science packages within a language, to more general topics like version control, documentation, and best practices in software engineering. Courses are typically delivered in the style of a workshop, with tutors teaching by example and frequent breaks for practise with exercises.

Short courses like the ones offered by RCDS or the Carpentries work well to equip learners with the fundamental skills to get started with a language or a package. As learners move beyond the basics, however, these short courses become a less effective and are replaced by other approaches [4]. In both research software, and general software engineering training, real-life projects and code examples are favoured approaches for supporting the transition from novice to proficient [5].

This paper introduces the ReCoDE[4] (Research Computing and Data Science Exemplars) project at Imperial College London, led by the RCDS team and supported by the central Research Software Engineering[5] (RSE) team. ReCoDE is a collection of guided, well-documented code examples based on the actual research of PhD students at Imperial, open to the public for independent study. PhD students pitch exemplar ideas in a rolling open call, and those selected undertake paid part-time work,

[1] imperial.ac.uk/students/academic-support/graduate-school/professional-development/doctoral-students/research-computing-data-science

[2] imperial.ac.uk/about/leadership-and-strategy/provost/vice-provost-research/vpre-led-initiatives/early-career-researcher-institute

[3] imperial.ac.uk/students/academic-support/graduate-school/professional-development/doctoral-students/research-computing-data-science/courses

[4] imperial.ac.uk/admin-services/ict/self-service/research-support/rcs/service-offering/research-software-engineering/case-studies/recode

[5] imperial.ac.uk/admin-services/ict/self-service/research-support/rcs/service-offering/research-software-engineering

collaborating with an RCDS and RSE team member over three months to develop their exemplar. The output of this collaboration is an online resource to guide students through a research software project, demonstrating best practices in data science and software engineering.

The major contribution of this work is to share the processes and technologies that support a resource like ReCoDE, whose aim is to source and share high-quality exemplars as learning artifacts. The next section contains a high-level overview of similar efforts towards collecting and sharing research software training resources. We then describe and detail the ReCoDE process as it exists today, which involves selecting, developing and publishing exemplars, followed by a results section discussing evaluation steps. We reflect on how our processes might evolve to provide a starting point for a more general framework for curating user-generated content in other educational settings. Finally, we conclude with our plans for the future of ReCoDE.

1.1 Related Work

In 2009, Stelma [6] writes about an online resource for PhD students at the University of Manchester. Within this resource, students can read first-hand accounts of previous students' experiences working with research software to learn from the examples of others. While this is an example of using research software rather than creating it, it captures the themes of sharing and learning by example and provides a good starting point for the discussion in this short section.

Developing software is a skill, and skills are acquired by doing, more so than they are by observing [5]. The cognitive apprenticeship model has been proposed for teaching and learning in several different disciplines [7], and it is a good fit for the skills of research computing, data science and software engineering. The apprentice learns by engaging with and following expert demonstrations of the approach, design and process of working towards a solution. The thinking processes of the expert are revealed to the learner, so that the learner may apply those processes in their own work.

This model of cognitive apprenticeship is reified in computer science and software engineering education as worked examples. These examples provide the expert step-by-step process to a solution, and they are considered an effective tool in teaching and learning [8]. In research software training, one paper proposes bigger, real-life projects to support learners with making the jump from taught lectures to practical implementation [5]. However, there is no mention of modelling expert behaviour to support the learner with the required skills for working their way through to the solution.

At the Fred Hutch Cancer Center, there were efforts to combine the ideas of worked examples and real-life projects into a repository of code examples that could be submitted by The Hutch research community [1]. The expert presence in these examples fits well with the model of cognitive apprenticeship, however the number and scope of code examples are currently limited. Also absent from this work is a process to ensure consistency, quality and best practice across code examples, to ensure that expert behaviour is truly being modelled.

The ReCoDE project is driven by the cognitive apprenticeship model and aims to bring together the advantages of worked examples with the benefits of working on a real-life research project. Sourcing exemplars from an ever-changing group of PhD students will ensure a consistent flow of new exemplars, which keep pace with the speed of cutting-edge research. ReCoDE proposes a thorough process for ensuring excellence in documentation that supports the learning process, and best practices that model expert behaviour. This process is described in the following section.

2 METHODOLOGY

This section focuses on the end-to-end creation of an exemplar for ReCoDE. A complete exemplar is typically a set of webpages explaining the reasoning, problem-solving and in some cases, troubleshooting, involved with a complete research software project. This can be in the form of code notebooks, such as Jupyter, or as tutorial-driven documentation. Exemplars do not aim to teach the basics of a programming language or tools but focus instead on their implementation. Discipline-specific material is minimised as much as possible to make exemplars accessible to those outside the specialisation. Particular attention is paid to what decisions were made on a design level during the production of the code, allowing the reader to follow and learn from the thought process of the author. The reason for the usage of a given tool is explained with the intention of illustrating to readers how tools they have learned about in abstract terms can be used in practice.

The below sub-sections detail the steps and processes that ensure the published exemplar is of interest to learners, has clear learning objectives, adopts good pedagogy, and follows best practices in software engineering for high quality code.

2.1 Selection

Ideas for exemplars are proposed by current PhD students at Imperial College London. There is a *Contributing*[6] section on the *ReCoDE Home*[7] page to provide essential information for students considering a proposal. This includes motivations for the project, elements of a good exemplar, and logistical details such as time commitment and pay.

The RCDS team coordinating the ReCoDE project aim to run four selection rounds per year, where exemplar proposals are scored, ranked, and assigned to team members. The scoring is based on the following four criteria:

- Novelty. Considering if there is already a published ReCoDE exemplar that uses the programming language, tool or library in a similar way.
- Topic. Considering if the topic is widely applicable across a discipline or disciplines, and not specific to a single niche research area.
- Scope. Considering if the exemplar is concise, both in terms of the size of the codebase and the number of learning outcomes.
- Resource. Considering whether the technical skill and person hours to support development are currently available in the RCDS and RSE teams.

After scoring, proposals are allocated to RCDS tutors based on their preferences. A similar allocation takes place within the RSE team. The PhD student proposing the exemplar is then notified of the outcome. Once these administrative tasks are complete, the exemplar development team is formed, consisting of the student developing the exemplar, the RCDS tutor, and the assigned RSE.

2.2 Development

The development stage begins with an initial meeting of the exemplar development team. The exemplar developer can explain their proposal in more detail, and the RCDS and RSE team members can ask questions or raise concerns. A recurring weekly meeting is scheduled for the following three months, and the developer is given two initial tasks.

First, the exemplar developer is asked to complete a project plan to firm up the learning outcomes, describe in detail the flow of a learner through the exemplar, and draft a week-by-week timeline of development. This project plan is shared with the RCDS and RSE team members for sign-off.

Next, they are directed to the *Developing*[8] section of *ReCoDE Home*, where they are guided through the process of setting up a repository in the Imperial College London enterprise GitHub and configuring it to automatically publish a GitHub Pages site with Material for MkDocs[9] using GitHub Actions.

MkDocs is a popular static site generator, and Material adds a modern theme as well as additional customisations and features. It is a popular choice for online software documentation and given that our exemplars are a form of extended documentation, it is a good fit for ReCoDE. Material for MkDocs supports a rich and extensible flavour of Markdown and can render Jupyter Notebooks directly. We do encourage exemplar developers to consider other ways to generate their documentation pages when appropriate. Some examples of other tools we have seen include Doxide for a C++ exemplar, Sphinx for a Python exemplar, Quarto for an exemplar in R and Documenter for a Julia exemplar.

The RCDS team maintains a template GitHub repository, where Material for MkDocs and the GitHub Actions for publishing to GitHub Pages are all pre-configured. To get started, the developer changes a handful of repository-level settings and can then begin to push Markdown files or Jupyter notebooks to the repository. When changes are registered in the repository, the GitHub Action is triggered, and the live site is updated to reflect those changes.

[6] imperialcollegelondon.github.io/ReCoDE-home/contribute

[7] imperialcollegelondon.github.io/ReCoDE-home

[8] imperialcollegelondon.github.io/ReCoDE-home/developing

[9] squidfunk.github.io/mkdocs-material

With the project plan agreed on by all members of the exemplar development team, and the GitHub repository configured, the development work can begin. Typically, the RSE will propose a software development methodology to adopt. This can vary from team to team depending on the experience of the developer and the amount of support they require. Teams will use a mix of project management and professional software development practices, from pull request reviews, to GitHub issues or project task-boards. Issues and code reviews are used to identify bugs, code smells and style issues, or to make suggestions on the documentation, code annotations, or other areas where teaching is taking place.

Weekly meetings provide the opportunity to discuss any larger, ongoing issues that could not be solved offline, and review the agreed development timeline. This weekly cycle continues until the exemplar is ready for a final review, typically after about 40-hours of developer work. In the final review, the RCDS tutor checks over the pedagogical approach and teaching materials of the exemplar, ensuring that the defined learning outcomes have been met. The RSE carries out a final check for consistency in style, a balance between readability and efficiency, and other aspects of software engineering best practices. With the final review complete, the exemplar is ready for publishing.

2.3 Publishing

Publishing a ReCoDE exemplar involves adding it to the list of currently available exemplars on *ReCoDE Home*. *ReCoDE Home* is itself a static site, built with Material for MkDocs and deployed via a GitHub Action to a GitHub Pages site on the Imperial College London enterprise GitHub. In addition to the *Contributing* and *Developing* sections, there are also sections for the *People*[10] who have contributed to ReCoDE, one with an interactive listing of the *Topics*[11] covered by the exemplars, and a section for the *Exemplars*[12] themselves.

When an exemplar is ready, the exemplar developer will reach out to the ReCoDE coordinators, providing photo assets for the people and exemplar pages, and a list of topics covered by the exemplar. Topics typically include the programming language, libraries, packages, or tools, and may include the scientific discipline, or a specific software engineering best practice that is emphasised in the exemplar. One of the ReCoDE coordinators will then publish this information to *ReCoDE Home*, linking to the GitHub Pages site of the developed exemplar.

3 RESULTS

There are several aspects of ReCoDE that are worth evaluating, both from the point of view of the exemplar development team, and of the learners who will be engaging with the exemplars.

Exemplar developers receive instruction in the methodologies, tools and approaches used during the development lifecycle, exposing them to professional software development practices and teaching approaches. The RSE and RCDS team members are exposed to a variety of scientific applications through the collaboration and gain experience of supervising a project. Anecdotal feedback during the pilot phase of ReCoDE has been positive. Going forward, each member of the exemplar development team will be asked to complete a feedback form, collecting data on what went well, and what could have been improved.

The real success of ReCoDE will be measured by the learner experience. After the first round of exemplar development, the ReCoDE coordinators ran focus groups with several PhD students external to the project. These students were beta testers, asked to engage in full with one of the exemplars as a learner. In the focus group sessions that followed, we collected student views on the experience. These included practical aspects such as presentation and accessibility, technical aspects such as difficulty and complexity, and others such as utility and range of activities. Focus groups discussed how learning with ReCoDE compares to other ways of learning research computing, and whether students felt such a resource was appropriate for bridging the gap between learning in a workshop and approaching a real research problem.

Overall, the sentiment in these focus groups was positive, with several students noting that the longer length, increased complexity, and detail of these exemplars provided more opportunities for learning than did workshop exercises, or online tutorials, with one student noting: “Many online tutorials hide complexity in functions. ReCoDE exposed everything.” Several students expressed their desire for

[10] imperialcollegelondon.github.io/ReCoDE-home/people

[11] imperialcollegelondon.github.io/ReCoDE-home/tags

[12] imperialcollegelondon.github.io/ReCoDE-home

having such a resource when they first started their own PhD journey. Some general areas for improvement were identified, such as harmonising the structure of the exemplars, improving signposting to documentation of the tools and libraries used, and improving the background and motivations behind each exemplar.

These early focus groups were helpful for informing our future evaluation process. The questions asked and discussion points raised are being used to create a feedback form that will be presented on each exemplar page, and on *ReCoDE Home*, to capture learners' experiences with specific exemplars, and with the resource as a whole.

One of the challenges of ReCoDE will be building an audience of learners to engage with the exemplars, and to become a resource that academic and research staff point students and researchers towards. ReCoDE is linked from the RCDS website, and the team can point to relevant exemplars at the end of our taught courses. To reach a wider audience, the team is developing an advertising campaign for the project, led by two PhD students in student-partner roles. We currently track visits to *ReCoDE Home* using Google Analytics and can track page exits to see which exemplars are being visited by how many learners. When our advertising campaign launches, we expect to see an increase in the number of visits to *ReCoDE Home* and see a balanced number of visits to all the individual exemplars in our collection.

4 CONCLUSIONS

At the time of writing, ReCoDE features over twenty published exemplars, covering five different programming languages, a multitude of libraries, packages and supporting tools, and several best practices, all across a variety of disciplines. There are six exemplars currently under development and another selection round coming later in the year.

The ReCoDE process defines a light framework, which we hope can serve as inspiration for educators in other STEM disciplines looking to source examples of good practice in their area. Running the selection phase as an open call generates interest, and we are always oversubscribed. A development phase that involves experts – in our case the RSE and RCDS tutors – benefits all those involved and ensures that best practice is integrated. Publishing with open-source tools keeps hosting costs minimal, and a formal evaluation process ensures that aims are met and that the resource is having the desired impact. There is much for us to refine and formalise in ReCoDE, just as there is much to learn from these early experiences.

We plan to formalise the publishing process further, injecting a more formal peer-evaluation step between the end of development and the start of publishing. Exemplar developers will have the opportunity to provide feedback on another exemplar that was created during their development round, and we will provide a structured feedback form to collect this data.

As the number of exemplars grows, navigating to an exemplar that can best support a learner becomes more difficult. As such, team members are currently re-designing the *ReCoDE Home* user interface to provide better experience. This will include search and filtering functionality, giving visitors the opportunity to browse by programming language, library, tool, or discipline.

As ReCoDE exits the pilot phase, we are focused on improving our feedback and feedforward mechanisms and further formalising our processes to ensure a high level of consistency and quality in all the developed exemplars. Most importantly we are focused on finalising and launching our advertising campaign at Imperial and beyond, to increase our user base and support learners with making the transition from learning a language or library to proficiently applying them in practice.

ACKNOWLEDGEMENTS

ReCoDE was funded by the Imperial College London Pedagogy Transformation Fund. We would like to extend a special acknowledgment and thank you to the Imperial Research Software Engineering team for being an integral part of this unique collaboration. A final thank you also goes to the exemplar developers and student partners, without whom ReCoDE would not be complete.

REFERENCES

[1] M. Barker, E. Breitmoser, P. Broadbent, N. P. Chue Hong, S. Hettrick, I. Lampaki, A. Quinn, and R. Taylor, "Software and skills for research computing in the UK," Zenodo, 2024. Retrieved from https://doi.org/10.5281/zenodo.10473186

[2] G. Wilson, "Software Carpentry: lessons learned," F1000Research, vol. 3, 2014.

[3] S. Crouch, N. C. Hong, S. Hettrick, M. Jackson, A. Pawlik, S. Sufi, L. Carr, D. De Roure, C. A. Goble, and M. Parsons, "The Software Sustainability Institute: changing research software attitudes and practices," Computing in Science & Engineering, vol. 15, no. 6, pp. 74–80, 2014.

[4] K. Hertweck and C. Strasser, "From novice to expert: Supporting all levels of computational expertise in reproducible research methods," in Practice and Experience in Advanced Research Computing, vol. 2020, pp. 407–413, 2020.

[5] K. Gallagher, R. Creswell, B. Lambert, M. Robinson, C. Lok Lei, G. R. Mirams, and D. J. Gavaghan, "Ten simple rules for training scientists to make better software," PLOS Computational Biology, vol. 20, no. 9, e1012410, 2024.

[6] J. Stelma, "An ecological model of developing researcher competence: The case of software technology in doctoral research," Instructional Science, vol. 39, pp. 367–385, 2011.

[7] A. Collins, J. S. Brown, and S. E. Newman, "Cognitive apprenticeship: Teaching the crafts of reading, writing, and mathematics," in Knowing, learning, and instruction, pp. 453–494, Routledge, 2018.

[8] B. M. McLaren, T. Van Gog, C. Ganoe, M. Karabinos, and D. Yaron, "The efficiency of worked examples compared to erroneous examples, tutored problem solving, and problem solving in computer-based learning environments," Computers in Human Behavior, vol. 55, pp. 87–99, 2016.