As software has become more essential to research across disciplines, and as the recognition of this fact has grown, so too has the importance of professionalizing the development and maintenance of this software. The community of software professionals who work on this software have come together under the title “research software engineer” (“RSE”) over the last decade. This, in turn, has created the need to understand how RSEs come into this profession and into these groups, how to further promote this career path to potential members as well as the need to understand what training gaps need to be filled for RSEs coming from different entry points. We have categorized three main classifications of entry paths into the RSE profession and identified key elements, both advantages and disadvantages, that should be acknowledged and addressed by the broader research community to attract and retain a talented, diverse pool of future RSEs.

Software has grown as a key part of research, along with digital computers, since the 1940s. Software developers originally came from the mathematics field, learning to program as needed, and then as computer science (CS) and software engineering developed, these fields began to develop standard curricula and training, leading to professional software development practices. Although the initial software developers came from a research environment, in part because all computing was initially research, as computing became more common and more widely used in business, programming and software engineering also became more formalized and professionalized. This led to a dichotomy between software engineering as taught and used in business settings, where it has been a profession, and as used in research, where it has often been one task among many performed by researchers, typically in universities and national laboratories, but also in industry in some cases.

In 2012, the lack of a professional role for software developers in research came to a head, and a breakout session in the 2012 Software Sustainability Institute Collaborations Workshop focused on common issues found by such software developers. The work done in this session and shortly thereafter created and defined the terms research software engineer (RSE) and research software engineering and began a movement that has now grown into a community with almost 10,000 members globally, multiple annual conferences and workshops, at least nine national and multinational associations and Societies, and formal RSE groups and career paths in many universities and national laboratories.

The authors of this article represent cofounders of the U.S. Research Software Engineer Association (US-RSE) and two of the largest RSE groups in the United States, 32 RSEs at the National Center for Supercomputing Applications (NCSA) at the University of Illinois at Urbana-Champaign, and 18 RSEs at Princeton University. These three activities have seen tremendous growth: US-RSE started in 2018 and now has almost
1300 members, the RSE team at the NCSA Software Development team had five members in 2012 and now has 32, and the central RSE group at Princeton started in late 2016 with a single RSE and will grow to 28 full-time staff in 2023. As leaders of these efforts, we also see a number of challenges, including a lack of general awareness of

- RSEs and RSE issues from stakeholders outside the RSE community, including many university administrators, many research funders, and many research publishers
- possible RSE roles by potential RSEs, including secondary students, undergraduates in both CS and other fields, and graduate students in CS and in computation and data-focused disciplines.

Although there has been progress in raising awareness in these groups over the past 10 years, the authors estimate, based on their experience, that the awareness is at best roughly 5% for some groups such as funders, and probably much smaller for potential RSEs.

This leads to two issues that diminish the amount and quality of software used in research:

1) The lack of general awareness and support disincentivizes those who are RSEs and are trying to grow RSE groups.
2) The lack of awareness by potential RSEs leads to lower-than-desired pools of candidates for existing positions, and when potential RSEs do discover this career path, they often aren’t well prepared for these positions.

Because, as of now, there are no formal educational programs specifically designed to prepare RSEs to enter the profession, RSEs emerge from multiple demographics. In the “RSE Career Entry Points” section of this article, we distinguish between a number of different entry points for different groups of potential RSEs, and discuss associated challenges, including awareness and preparation.

**RSE CAREER ENTRY POINTS**

The term RSE can be used to broadly classify individuals who use software engineering to advance research. This broad classification means that a qualified candidate might have many different educational backgrounds and experiences. It also means, however, that the specifics of a particular position often substantially define the requirements of a role. Nevertheless, a handful of overarching categories tend to emerge, each with its own strengths and weaknesses. We want to highlight that because the RSE profession is still relatively new, there are a number of challenges associated with each entry point, but these shouldn’t be seen as outweighing the advantages that we have clearly witnessed. Rather, our hope is to identify the key challenges that current entrants to the profession are likely to face in an effort to focus community efforts on minimizing them. By providing avenues to reduce these challenges, we stand to improve and diversify the RSE pipeline and better retain existing RSEs.

We discuss the three top-level categories that capture most RSE experiences: a domain science background, a pure CS background, and industry experience as a software developer. We also recognize that there will be cases where an individual might fall into two categories simultaneously, for example, a physics Ph.D. graduate who worked in industry for three years as a software developer before taking his or her first RSE position. These special cases can ameliorate some of the challenges associated with one of the categories individually, but won’t be specifically addressed here.

**DOMAIN SCIENCE**

According to the 2022 RSE survey, more than 75% of U.S. respondents identified with an educational background of something other than CS. Similarly, more than 50% of all respondents had listed a Ph.D. degree as their highest level of education. Clearly, formally trained researchers in domain sciences is one of the largest entry points to the RSE profession. This demographic is consistent with the authors’ observations. In this category, we include both recent graduates and those with additional experience, either in a nonacademic setting or as a researcher, such as a postdoctoral researcher (postdoc). Often, these RSEs are people who started in a particular discipline but discovered they enjoyed the software aspects more than other aspects and possibly wanted to broaden the application of their software skills.

Entering the RSE profession with a domain science and recent research background (i.e., as a recent Ph.D. graduate or postdoc) has a number of benefits that are immediately applicable to an RSE role. First, the new RSE has sufficient understanding to digest new research problems and communicate effectively with non-RSE researchers. Second, he or she understands the research culture, goals, and incentives, making new collaborations with researchers at times easier as they speak a shared language. Additionally, those self-selecting to enter the RSE professional have typically learned software development skills independently, thus developing the capacity to self-learn new technologies and approaches.
RSEs with a research domain background potentially interested in an RSE or RSE-like role will often face four main challenges: 1) awareness, 2) technical preparation, 3) career prospects, and 4) crossing disciplines.

Because the RSE name and career path has only recently seen an increase in exposure and publicity, many researchers fail to realize that the role is becoming mainstream. It is the authors’ experience that upon learning of the RSE role, graduate students often become excited at the prospect of doing RSE work. If potential qualified individuals are not even aware that a role exists, they will not be searching for openings and therefore fail to enter the RSE pipeline.

Former researchers from backgrounds other than CS are frequently faced with learning software development on their own, without formal training or mentorship. Although this can be extremely effective, it can also leave gaps in knowledge unknown to the individual. It can also lead to imposter syndrome.

Many university RSE positions could be viewed as less desirable to other career opportunities, often seen by primary faculty as a lesser position that is not doing research, only supporting it and lacking the possibility of tenure. Positions are often grant funded, and therefore limited to a fixed term. It has been our experience that many former researchers are interested in job stability and specifically not having to tirelessly pursue additional funding.

It can be scary for researchers with a Ph.D. degree to leave their own discipline and learn others. Exposing the advantages of an RSE position, including the opportunities to use existing knowledge in new areas and to learn and contribute to new disciplines, can overcome this.

**EARLY-CAREER CS**

As mentioned earlier, more than 75% of RSEs have a background in a non-CS field. That means, however, that nearly 25% of RSEs have a background in CS, larger than any other single field. Here we discuss early-career professionals with a formal CS education. We define early career as up to three years of postundergraduate (undergrad) working experience. Beyond three years of experience, we would consider an RSE with an undergrad degree as a different entity, for example, an industry software engineer (see the “Industry Software Engineers” section).

Considering a typical RSE requires a mix of research and software engineering, a CS undergraduate degree is likely the best formal education to prepare for the software engineering perspective. Additionally, many graduating students aren’t yet sure of their long-term career goals and seek to further their skills in some way, perhaps getting a master’s degree, or as we have seen, working in academia as an RSE for a few years.

As with the previous entry point, undergraduate students are unlikely to be aware of the RSE career path and perhaps even with research in general. Without exposure to graduate-level research, the concept of research software is more abstract. Therefore, they are unaware of the potentially interesting, societally relevant, and intellectual challenging projects associated with research software engineering.

Like all entry-level and early-career professionals, a new RSE needs a significant amount of supervision and guidance. The aspects unique to RSEs in this category (as opposed to all entry-level positions) that seem to cause the most problems stem from elements inherent to the research workflow. This includes 1) the often nebulous and vague requirements of research software, 2) the incentives and priorities of researchers, and 3) the variation and inconsistency between projects.

Some CS students may not learn software engineering or how to apply it. In this case, CS can be seen as similar to any other discipline. However, the number of computing courses likely means that these RSEs have had some exposure to some software engineering practices, particularly at universities that emphasize using common tools and practices to prepare their students for jobs in industry.

**INDUSTRY SOFTWARE ENGINEERS**

A growing entry point into the research software engineering field is from those with experience working as a software engineer in a nonresearch environment, which we’ll call industry. Clearly, there are RSEs at companies and private industry working as part of a research project, however, for this classification we’re going to use the term to refer to those who are developing software for applications other than research.

Exposure to best practices, rigorous software engineering, and working as a part of larger teams are some of the clear benefits of having industry experience. Much of this expertise is transferable to the RSE role. In particular, software consultants from industry who also have hands-on experience are particularly valuable as they have done work similar to that of senior RSEs, although perhaps not in a research context.

The biggest challenge with RSEs moving from industry to a research environment is the stark contrast in culture, priorities, and development environment. Many, if not most, nonresearch organizations are driven by business profit and, as a result, software
development practices reflect the business need. Strict, often unwavering, best practices must be followed at all times. Software engineers often work on software teams, with ownership only over small pieces. Software engineers are often removed from clients and customers relying on project managers to relay requirements. In an academic setting—where requirements are often vague and change quickly and timelines that have lulls in oversight—sudden, fast-paced needs around things like conference and journal deadlines or annual reviews can be frustrating to software engineers from industry. Another challenge is that of motivation. Many who take the route of getting a degree in CS are motivated not only by an interest in the field but by the high monetary salaries of industry positions. This is typically not the case in academia, however. On the flip side, academia provides a good work-life balance as compared to many industry positions, especially those in start-ups or fintech, often very good benefits with significant vacation time, interesting cutting-edge problems that can potentially benefit all of humanity, and the ability to have direct input, leadership, and ownership of the work. A good number of RSEs who come from industry gladly make the trade, wanting to be more than a cog in a massive machine, but pursuing recognition and impact, making a difference with their work.

The authors’ have observed a tendency among some RSEs coming from industry to not fully appreciate key aspects of research, in particular, they often don’t have or don’t see the need for the equivalent of a literature review (i.e., knowing the current landscape, what has been done already, and so on). This awareness of the field, however, is critical to truly drive innovation and impact and not reinvent the wheel.

OTHER NOTABLE ENTRY POINTS

We believe that the three categories capture a significant fraction of the current entry points into the RSE profession. Others not specifically addressed include

- **Recent M.S. graduate**: This is something of a mix between the first two main entry points: domain science and entry-level CS. Individuals in this category have more experience and research expertise than those who are entry level but less science and research experience than a Ph.D. graduate.
- **Professional research staff (including domain scientist or computer scientist)**: Individuals in this category are even more experienced and mature than new Ph.D. domain scientists but are often so entrenched in the research process that they struggle with the transition to prioritizing software engineering aspects.
- **Data scientist/engineer**: Data scientists often employ many research software engineering approaches. Moving to a data-heavy RSE project can be a smooth and logical transition.
- **Research computing facilitator or research systems administrator**: Those in these positions are often adjacent to both RSEs and researchers, giving them a clear understanding of research incentives and processes. In some cases, individuals in both of these jobs may already have an element of research software engineering as part of their regular job duties.

PROPOSED ACTIVITIES

As the RSE profession expands and garners national and international attention, the need to build a diverse and prepared pipeline and RSE workforce is becoming increasingly important to the long-term health of the profession. With the identification of the main entry points, we can begin to address the needs of each demographic. Here we propose specific activities to facilitate the preparation and success of new RSEs.

AWARENESS

Although a relatively large number of individuals now identify as RSEs (see Figure 1), we believe that this is only the tip of the iceberg, with many software engineers in academia still unaware of this effort, without career paths, isolated and scattered across university projects, and living from grant to grant. This seems to stem from not only a lack of awareness from academic software engineers but also from university leadership, despite the fact that those in leadership positions at major research universities are often working to address an ever-increasing need for software, data, and computation as part of their research missions going forward. Although many people today acknowledge the need for better, bigger, more extensible, and sustainable research software, the means to do this is still too often hiring students, encouraging researchers to do software development, tasking the campus IT staff with such development, or attempting to hire an external software firm and providing them with the software’s requirements if possible. As groups such as US-RSE, the Research Software Alliance (ReSA),d the Academic Data Science Alliance,e and so on gain

\[\text{https://us-rse.org}\]
\[\text{https://researchsoft.org}\]
\[\text{https://academicdatascience.org/}\]
momentum, effort must be made to not only increase awareness of this RSE profession and how it benefits the research enterprise among software engineers in academia but also among university leadership. As outlined in the “Other Notable Entry Points” section, entry points into the RSE profession are varied, but because the RSE career path is still in its infancy and there are no formal RSE education programs, word of mouth, professional networks, and in some cases, luck, are often the catalyst for future RSEs to learn of the profession. As a result, the potential pool for RSEs is often limited to those fortunate enough to stumble onto the concept, or those with significant professional networks. This limits the diversity and breadth of an RSE pipeline. We believe that a concerted outreach effort by the existing community, with assistance and acceptance from the established research/educational community, is the best approach to expose potential future RSEs to the concept.

Given our past experience, outreach activities to students of all levels that both introduce the concept of RSEs and legitimize the career have been successful. For example, seminars and presentations, both in person and online, that explain RSE work with examples of actual RSE projects, are frequently met with excitement and enthusiasm from students who were previously unaware. The authors have received positive feedback from high school, undergraduate, and graduate students who have attended outreach talks.

Established RSE group leaders are in a unique position to give such presentations and should be encouraged and incentivized to do such outreach activities wherever possible.

US-RSE was founded in early 2018, and in the following four years has grown to more than 1300 members (see Figure 1). We believe that US-RSE is poised to make a significant impact on the future of the RSE profession. Many of the items we list here are directly tied to US-RSE’s mission to create a community, advocate for RSEs, provide resources in support of RSEs, and promote, encourage, and improve diversity, equity, and inclusion within the RSE profession.

US-RSE is a part of the larger global RSE community, which started with the U.K. Research Software Engineers Association, which has now turned into the Society of Research Software Engineering, US-RSE, the Society, and other national and multinational RSE organizations work together on common activities, including on RSE recognition and promotion. In addition, ReSA and national organizations that are working toward software sustainability, such as the U.K. Software Sustainability Institute (SSI) and the U.S. Research Software Sustainability Institute (URSSI), work closely with RSE organizations as they understand that having a strong

---

**FIGURE 1.** Number of members in the US-RSE organization. (Source: https://us-rse.org/join/.)

---

1https://www.software.ac.uk/
2https://urssi.us/
RSE community benefits the overall research software community.

National RSE associations, such as US-RSE, are increasingly positioned to organize and publicize outreach activities that bring awareness to students and other early-career individuals. US-RSE should continue to produce events such as early-career panels, publish examples of RSE work, and facilitate sharing of outreach activities and approaches.

**TRAINING AND EDUCATION**

There is growing recognition among the scientific community for what are being called *cyberprofessionals*, of which RSEs are under. A major step on this front occurred in 2021 with the inclusion of a cyberprofessional mentorship plan as part of proposals such as for NSF’s Cyberinfrastructure for Sustained Scientific Innovation (CSSI) program, requesting information as to how such staff are mentored, maintained, and so on. Additionally, solicitations continue to emerge around the training of cyberprofessionals, such as the CyberTraining call, with efforts spanning research experiences for undergraduate opportunities for students to work with experienced cyberprofessionals as an introduction to this area, to supporting the training of researchers and software engineers in particular areas with regard to emerging technologies. The CyberAmbassadors effort, for example, funded through the CyberTraining program for nearly five years now, has been very useful as a tool for introducing new cyberprofessionals to successfully working within an academic setting. However, although we are beginning to see training and education opportunities for RSEs, there is still much more to be done, as described next.

(Research) Software Engineering

To date, there are no formal accredited courses or programs in research software engineering. As mentioned previously, new RSEs coming from a domain science often lack exposure to rigorous training or education on software engineering concepts. Easy access to targeted training material, workshops, and courses would enable those coming from non-CS backgrounds to fill the gaps in their knowledge and accelerate their self-learning. Although a number of software engineering bootcamps and online courses exist, few target research software-specific issues and concepts at a level necessary for adequately preparing RSEs. Better Scientific Software,\(^6\) INTERSECT,\(^7\) and CodeRefinery,\(^8\) are examples of projects and initiatives making progress in this area, but more work is needed.

Because technical preparation is a major challenge for multiple RSE entry points, a formal degree program, for example, a master of science (M.S.) degree in research software engineering would not only help domain scientists and recent CS graduates gain much-needed experience but would also provide visibility and awareness of the RSE career path. It is our opinion that an M.S. degree in research software engineering would be a popular degree program and teach skills applicable for professions other than just research software engineering. In addition to RSEs, academic researchers who write software and industry software engineers would be reasonable and appropriate careers for graduates from an M.S. degree program.

Research Fundamentals and Domain Science

To address the need for learning research fundamentals for those without a domain science research background, targeted domain-specific programs would teach those new to the field the basics of research incentives, culture, and processes. This would not be meant to re-create a graduate degree but rather prepare new RSEs to acquire the skills necessary to self-learn the elements of the domain to collaborate effectively with domain researchers. More work is needed in this area, both to identify relevant topics as well as to develop and deliver such domain-specific educational programs. This is another area where national RSE associations, such as US-RSE, can and should take a leadership role in providing a forum for focused discussion and the creation of an educational framework that multiple domains could leverage.

Software Development Versus Maintenance and Reuse

Because aspects of academia encourage creativity and innovation, there is a tendency at times to think any self-devised solution is novel by default and will solve a breadth of scientific challenges without first looking at what others have done (a time-consuming task in and of itself).

Specifically with research software, a common problem is the endless reinventing of the wheel. This leads to terms such as *yet another workflow system* or *yet another data management portal*, systems that have most of the same capabilities as those that already exist, but one was made by biologists while another was made by geoscientists who were unaware of the others’ work. RSEs need to have a mindset of starting with a

---

\(^6\)https://bssw.io/
\(^7\)https://intersect-training.org/
\(^8\)https://coderefinery.org/
literature/software review, not only on the science side but also on the research software/cyberinfrastructure side to understand whether there is really a need to develop new software, or whether it would be better to use existing software, perhaps adding a small number of features to it.

**Internship and Mentorship Programs**

Our experience with our existing RSEs has been that the fastest and most effective way for new and aspiring RSEs to learn the necessary skills and knowledge is through formal and informal internship and mentorship programs. Even if more formal training avenues exist, research software engineering is still a craft that needs practice and exposure. By working with a more experienced RSE mentor, a new or aspiring RSE can quickly learn applicable best practices and techniques, both in terms of technical skills as well as those social skills useful in contributing to team projects. This real-world experience is effective and frequently transferable to other projects and/or domains.

Organizations with large RSE groups can support such programs internally, however, small programs with only a handful of RSEs, or even just one, will struggle to provide internship and mentorship opportunities. It is our hope that the professional community, perhaps through national organizations such as US-RSE, can provide an accessible avenue for students and early-career professionals to match with an internship or mentorship opportunity, even if it’s located at a different institution or organization.

**Professional Support**

In the 10 years since the term RSE was coined, the profession has grown considerably, and despite the work that remains, awareness has increased. The profession, however, still struggles with stability and long-term career prospects resulting from its relative nascency in the research ecosystem. If our goal is to attract and retain talented and skilled professionals, we must make the profession desirable and appealing to join and remain in, not just for early-career professionals, but for mid- and late-career ones as well. This means having a clear and transparent career path with advancement opportunities. RSE position terms should be permanent and open ended, providing RSEs with desirable and stable employment. Because it’s human nature to want to be valued and recognized for one’s work, the entire research community can help by giving credit for RSEs’ contributions to research. Additional credit and recognition approaches, such as providing mechanisms for crediting software contributions, making software citable, and regularly citing software used, will help raise the stature of RSEs and recognize the increasingly important contributions they make. This, in turn, will help attract and retain our best RSEs.

**Conclusion**

Over the past 10 years, the idea of RSEs has been created and developed to recognize and promote a type of work that has long existed in scholarly research but has not had a uniform name or description. The RSE movement has built community among RSEs, many of whom didn’t know they were RSEs initially, and has also built community awareness, leading to RSE associations in multiple countries and RSE groups in many universities.

These formal groups have equally formal management structures, and one of the largest challenges for those who lead these groups is how to staff them. This involves finding qualified staff, either with the right software engineering skills or with other relevant knowledge and the ability to learn these skills. Additionally, to build this workforce, it’s important for the possibility of an RSE career to be widespread so that potential future candidates both are interested and develop a set of the necessary skills.

This then leads to a definition of how RSEs enter the field today and what training is needed. RSEs generally come from one (or more) of three types of background: CS graduates, typically with an undergraduate or master’s degree; Ph.D. graduates, typically from a computational or data intensive field; and software engineers with experience in industry. For each, we discussed the advantages and disadvantages they bring to an RSE position as well as how we might better expose them to the possibility of such a position, and how we might educate and train them for it.

We suggested a number of community needs, such as more activities to increase awareness, train and educate future RSEs, and support current RSEs through increased recognition. We believe that national organizations (for example, US-RSE in the United States) can and should play a significant role in both sponsoring such activities and providing the documentation and resources necessary to support the organization of local activities.

**Acknowledgments**

We would like to thank the overwhelmingly motivated and positive members of the both the United States and international RSE community for openly collaborating and supporting any and all RSE efforts. A special thanks to the early members of the Society for
Research Software Engineering (formerly the U.K. RSE Association) for starting 10 years ago what would become an international RSE movement. We also want to thank our own organizations, Princeton University and NCSA/University of Illinois, for their support of the RSE movement and our RSE groups.

REFERENCES

1. R. Baxter, N. C. Hong, D. Gorissen, J. Hetherington, and I. Todorov, “The research software engineer,” in Proc. Softw. Sustainability Inst. Collaborations Workshop, 2012.
2. D. S. Katz, K. McHenry, C. Reinking, and R. Haines, “Research software development & management in universities: Case studies from Manchester’s RSDS group, Illinois’ NCSA, and Notre Dame’s CRC,” in Proc. IEEE/ACM 14th Int. Workshop Softw. Eng. Sci. (SE4Science), May 2019, pp. 17–24, doi: 10.1109/SE4Science.2019.00009.
3. D. S. Katz, K. McHenry, and J. S. Lee, Jan. 2021, “Research software sustainability: Lessons learned at NCSA,” distributed by University of Hawaii. [Online]. Available: http://hdl.handle.net/10125/71494
4. “Princeton bets big on research software engineering,” Princeton Univ., Princeton, NJ, USA, Aug. 2022. [Online]. Available: https://researchcomputing.princeton.edu/news/2022/princeton-bets-big-research-software-engineering
5. S. Hettrick et al. International RSE Survey 2022. (2022). Zenodo. [Online]. Available: https://zenodo.org/record/7015772v.ZBMARhRB5c
6. A. M. Smith, D. S. Katz, and K. E. Niemeyer, “Software citation principles,” PeerJ Comp. Sci., vol. 2, Sep. 2016, Art. no. e86, doi: 10.7717/peerj-cs.86.

IAN A. COSDEN is director for research software engineering for computational and data science at Princeton University, Princeton, NJ, 08544, USA. Cosden received his Ph.D. degree in mechanical engineering from the University of Pennsylvania. His research interests include Research Software Engineering (RSE) training and education, RSE business and functional models, and career pathways and entry points for research software engineers. Contact him aticosden@princeton.edu.

KENTON MCHENRY is associate director for software at the National Center for Supercomputing Applications, the University of Illinois at Urbana-Champaign, Urbana, IL, 61801, USA, where McHenry received his Ph.D. degree in computer science/artificial intelligence. His research interests include data access, sharing, and publication; data analysis/machine learning; cross disciplinary frameworks, dashboards, and gateways; and tool preservation and interoperability. Contact him atmchenry@illinois.edu.

DANIEL S. KATZ is chief scientist at the National Center for Supercomputing Applications and a research associate professor in computer science, electrical and computer engineering, and the School of Information Sciences, the University of Illinois at Urbana-Champaign, Urbana, IL, 61801, USA. Katz received his Ph.D. degree in electrical engineering from Northwestern University. His research interests include policy issues including citation and credit mechanisms and practices associated with software and data, organization and community practices for collaboration, and career paths for computing researchers. He is a Senior Member of IEEE, the IEEE Computer Society, and ACM. Contact him atd.katz@ieee.org.