Retract bit-rotten publications: Aligning incentives for sustaining scientific software

James Howison
University of Texas
1616 Guadalupe Str, Austin, TX
jhowison@ischool.utexas.edu

ABSTRACT

How might we use our current publishing and academic reputation system to promote software sustainability? This paper seeks to provoke discussion by proposing that papers whose workflows are not kept current with the changing software ecosystem should be automatically retracted, placing the work of adjusting a purported contribution on those receiving the credit. This would be a change from the current situation where the work is placed on those seeking to use the contribution, each of which labors independently. As a provocation the proposal brings out strong reactions, and helps clarify what readers think of as worth rewarding in science. I progressively soften this proposal, eventually coming to a standpoint whereby a continuous improvement system could highlight opportunities for others to update a workflow to match changes in the software ecosystem, receiving a range of rewards from addition to a paper's author list, to in-document acknowledgment, acknowledgement on a publisher's or other website.

INTRODUCTION

Currently our publication system rewards authors at the time of publication. Ostensibly publication is the event that renders the material available for use by others (at least when complemented by availability through libraries). Yet there is a time-lag between publication and use of the contribution by others (perhaps the most telling assessment of impact). For many types of contribution this time-lag is not too problematic: abstract concepts such as mathematical proofs may not decline in usefulness, but for contributions reliant on software, the time gap between publication and attempted use can be highly problematic, as described below, creating a need for significant work before new science can be undertaken. For software, at least, publication, which generates the reputational reward, is removed from use, which generates a key scientific benefit.

In this paper I make a provocative proposal that envisions an actionable tweak to the publication system that re-aligns these elements while simultaneously opening up opportunities for education and legitimate peripheral participation in cutting edge scientific research. First, however, I build my argument by revisiting the oft-cited goal of replicability in scientific research.

The four meanings of replication

Much of the discourse on software in science emphasizes the scientific goal of replication, yet the meaning of that word is somewhat unclear. I argue that there are four distinct meanings in the discourse, each positive. First, replication implies that the scientific result in question can be repeated which implies the essential validity of the result. The second meaning is re-execution, by which the exact same software might be run to produce the exact same result. The third implies parameterization, by which the exact same software might be run with different inputs (say, data and/or configuration settings) to produce novel results. The fourth meaning invoked when we talk of replication is that a scientist might take existing work as a basis for new work, replicating in order to extend in some fashion. In these senses we see a rising contribution to the progress of science, extensibility being far more closely linked to progress than, say, mere re-execution.

Each of these practices are invoked when we speak of replication, yet seeking to achieve each has quite different implications for policy, particularly in a software context. It is entirely possible for a setup to achieve, say, re-execution without achieving parameterization nor extensibility. For example, an archived virtual machine that freezes the full software assembly (data and workflow components with their full set of dependencies) certainly achieves re-execution. If we assume that the code is available in source form, then the virtual machine likely also contributes to repeatability since it can be inspected and understood (the archiving assists in the transparency of scientific practice). Indeed it is only as a study object that such a frozen assembly of software necessarily contributes to the further meanings of replication.

Empirical studies of scientific practice show that scientists typically seek to replicate in order to extend . As such the scientist seeks to simply to run the code, but to work with the code. Needing to work with the code drives the scientist towards current, updated, versions of the software, both immediate workflow elements and their dependencies. Newer versions not only allow for new scientific methods, but also new features, better performance and new hardware support. Possibly even more importantly, communities that support

the software will often decline to help with older versions, either because they no longer remember or because helping with older versions means working around already fixed issues and takes away time (and user experience) that might help improve the current version (cite usability study).

The pattern described above also takes place at projects producing software components, each of which is also an assembly of directly used components and their dependencies. Each project works hard to keep their software in sync with surrounding components, adjusting their code to keep it current. In Howison and Herbsleb (2014)[1] we explore the category of maintenance and describe the three types of work this requires: sensing what is changing, adjusting to keep things running, and synchronizing with other projects to minimize the downstream adjustment work of others.

In summary, then, the most scientifically useful code is that which facilitates extension by others, and a key challenge for that is motivating the work required to facilitate extension. Indeed, since the initial production of software is often sufficiently motivated by the production of papers or provision of grant money, motivating the ongoing work of keeping things running and up to date with the state of the art in technology and science is core to the question of sustainable scientific software. How might we motivate such work?

The question is particularly difficult because we not only need to motivate it one-time but we need to motivate it for the long term, off into the relatively unknowable future. Further, we need to ensure that we are directing scientific effort towards the most needed software; an ideal system would motivate in proportion to the scientific usefulness and impact of the software in question.

A PROPOSAL

The proposal of this paper is simple and begins where others have left off in proposing improvements to publication for replication:

1. Papers should only be accepted if their full workflows, data and results are provided, enabling re-execution.

2. In addition authors should provide a set of regression tests and the software assemblage for each paper should be continually re-executed using a continuous integration system, such as Travis.

   So far these elements cover practices already adopted (e.g., at RunMyCode.org [3] and the Biostatistics Journal [2]), but the key extensions are:

3. The integration system would, prior to re-executing, pull each new version of dependencies, keeping each at the most current version and then re-execute the workflow and the test suite.

4. When a software assembly fails to execute, or begins to fail tests, the accompanying paper would be retracted by the journal.

Since a retraction is an event which threatens the reputation of its authors (being a greater negative than working on new papers is a positive), the authors of the paper will be highly motivated to investigate and correct the error, mostly likely making relatively small adjustments to keep the software assembly working. Those adjustments could then be pushed upstream and made available to everyone.

The beauty of this scheme is that it not only exposes the need for work to keep software scientifically useful, but it does so at the moment the need for the work becomes apparent, thus likely minimizing the required adjustment work. Moreover the scheme moves the burden of sensing, adjustment, and synchronization from the many, distributed, future users and centralizes it, reducing the aggregate burden. Finally the scheme motivates those most intimately connected with the code and thus best placed to undertake the adjustment work.

In this pure form the proposal synchronizes the earning of reputation from a publication from the technical debt of the work needed to ensure the ongoing usefulness of the claimed contribution.

Yet some may well feel (and not unreasonably!) this proposal is too onerous for the authors and, by creating an unknown future burden with each and every publication, might chill the very essence of scientific communication: maintaining transparency and openness that others might learn. Certainly the proposal fails to distinguish between those contributions that continue to be scientifically useful, given the cutting edge of science, and those whose time has passed. Thus we amend the proposal to suggest a restatement of the retraction rules:

4. On a failed re-execution, the paper would be flagged as "provisionally non-extendable" and the authors informed. Each copy of the paper would prominently feature this flag. They would have a reasonable period of time (36 months) to correct this issue and have the flag removed. After this period (or by choice of the authors) the paper would be tagged as "retired" and featured less prominently in the journal.

This creates a novel tier of publications, beyond the simple triple of "not published", "published", and "retracted". Authors and those responsible for tenure and/or promotion could highlight the number of papers that remain at the highest levels of contribution: those that provide an immediate base for the advancement of science by others through extension.

Yet the emphasis remains entirely on the original authors to maintain their contributions. If the system was successful and the opprobrium of having publications tagged as "retired" was great, then the same concerns about chilling effects would be valid. Worse, if the tag of "retired" was not considered dishonorable then the system would not improve things at all.

How might this issue be turned into an opportunity to further improve things? To address this we propose yet another restatement:

4. On a failed re-execution, the paper would be flagged as "needing work" and the community informed. Anyone, including but not limited to, the original authors, could then undertake this work, returning the paper to full sta-

tus. Those who did so would be added to the author list of the paper.

The beauty of this proposal is that it not only highlights the work that needs to be done, but creates a reputation market to attract those interested in undertaking the work. Moreover, since it is more valuable to be an author of a highly cited (and therefore arguably more scientifically useful) contribution, the system would provide greater rewards for more useful work, allowing effort to be concentrated on the most important contributions and allowing those of primarily historical or study value to recede to (and be marked with) appropriate status.

Yet some still might find the proposal too aggressive: after all authorship is a core value in science, one might even say sacrosanct, and the prospect of having unknown individuals added to a paper would quite likely chill participation. This reflects the fact that authorship reflects credit for many things in combination: initial ideas, hard work in data collection and only in part the focus of this proposal which is credit for providing a platform for others extensions.

Accordingly, one might experiment with a range of rewards for sustaining the extendability of scientific papers, with being added to the author list at the large end of the scale. Other options would include being added as an acknowledgement in the downloadable paper, perhaps in a new category (as some journals do by acknowledging the work of reviewers and editors). Options also exist that don't alter the original paper (and are therefore more palatable or incremental), such as including a coversheet that lists ongoing contributions. At the small reward end would be forms of acknowledgement that exist separately from the paper, such listing on the paper's landing page. Still further would be listing credit separately from the publisher and paper entirely; that would have the downside of having limited effect, but the upside of being implementable without publisher support.

Who might be motivated to achieve rewards across this spectrum? It is hard to tell in prospect; perhaps with the right celebration and featuring of these contributions many will be attracted, particularly those who seek to make their contribution through scientific infrastructure. Certainly, the status of "maintainer" of packages within linux distributions comes with considerable reputation and lucrative consulting opportunities.

Another, perhaps even more likely, source of motivated participants would be early stage researchers would be most attracted. For these potential participants these rewards could provide an entry point into the scientific reputation economy, constituting the kind of legitimate peripheral participation that creates an entry point to a community of practice [4]. Certainly these contributions would provide a new kind of quality signal while simultaneously motivating students to undertake replications. Since replications involving the nitty-gritty of the work are an excellent teaching tool (cite, cite) this proposal might creates a new contribution to education.

Would this sustain the right unit of analysis?

By focusing on software assemblies underlying domain scientific publications, this proposal shifts the emphasis from sustaining software packages to sustaining workflows and assemblies. Moreover it creates incentive for incremental adjustment rather than broad-based refactoring and as such might lead to increased work by sustaining (and even exacerbating) non-performant architectural choices.

It might, however, be possible to view the aggregation of adjustment work occurring within the publication maintenance system as a source to be monitored and even data-mined to identify opportunities for software package maintainers to collect and rationalize the adjustment work being undertaken. Such rationalizations would resolve issues before they show up in the publications; it would be interesting to consider how such work should be accounted for in the system. Certainly software component producers would have clear evidence of their impact: they would know where their components are contributing and, as others make adjustments in their components to sustain publications, would have evidence of impact. Finally as they merge those adjustments upstream, component producers would have evidence of the software stewardship they provide.

CONCLUSION

This paper is a proposal that we utilize the existing "currency" of academic life, publications and citations, to incentivize the work needed to sustain scientific software. We go beyond existing proposals that publications include executable software assemblies; these support re-execution (and transparency that enables reimplementation) but do not adequately support the more important goal of facilitating the progress of science by facilitating extension of others results. To achieve this we make the provocative suggestion that publications whose workflows are not maintained to work with the moving and changing software ecosystem should be retracted. We temper this by suggesting such publications be tagged and thus create an opportunity for a new type of scientific contribution, akin to package maintenance in linux systems: those that reanimate the workflow with the latest components are added to the paper, accessing credit for maintaining a scientific contribution in its best and most useful form. The proposal is systematic and actionable; moreover it just might work.

REFERENCES

1. Howison, J., and Herbsleb, J. D. The sustainability of scientific software production.
2. Peng, R. D. Reproducible research and biostatistics. *Biostatistics 10*, 3 (July 2009), 405–408.
3. Stodden, V., Hurlin, C., and Perignon, C. RunMyCode.org: A novel dissemination and collaboration platform for executing published computational results. In *2012 IEEE 8th International Conference on E-Science (e-Science)* (2012), 1–8.
4. Wenger, E. *Communities of Practice. Learning, Meaning, and Identity*. Cambridge University Press, New York, NY, 1998.