What are the current must-use standards for publishing and reproducible research?

What are the five most important things learnt during this discussion:

(Apologies for taking a different approach here. I think the diversity of thoughts and ideas are exactly what made this discussion so fruitful, so I'm hesitant to condense it too much)

Development in 2 years
More cross-disciplinary and interdisciplinary publication opportunities. Lots of experimentation.: Metadata around authors and papers for journals, type of contribution. Would like to see more first authorships/ownerships for PhD students, e.g. in biology. Bundled publishing in projects, better impact metrics. Louder shout about problems, creating more noise and more bureaucracy. The more you log and measure, the more bureaucracy and policying emerges. CS community starting to realize drawbacks of combinening travel (conference) with formal publication. Better curation of data and material associated with articles. Make it searchable and usable by anybody.

Development in 5 years
Funders may be requesting change in behavior from researchers. ID is not addressed well in academia, will have to improve. Some of the key funding programs are just starting now, hopefully will provide solutions. Impact = relevance: this should take relevance and reproducibility in account. Impact also needs to propagate faster. Also need improvements in accessibility. We need a policy group that guides the next REF, and advices on how to judge _all_ academic outputs. Perhaps less change than expected on this time frame, same people likely still in charge. Full generation needed to make substantial changes happen. Scientists not living up to the metric requirements may have been fired by then. Publish-then-review schemes. Review credit, better structured career development, bolstering collaborative research efforts. By this time we'll obtain an overview of policy, the power flows back to the people doing stuff and not caring about it. Various changes in existing people collaborations and new unexpected avenues of collaboration. Systems will have been integrated better, making it easier to collaborate and share research across the distributed ecosystem.

Development in 20 years
Publish anywhere using whatever mechanism gives me the best support for my writing. Not a marketing case, but a case for support. Scientists should be in charge of funding bodies. Publishing becomes a continuous process. Peer review on publication side, but also peer review from your personal peer reviewer. Development of more flat governments, and flat governance. Publish-then-review schemes will/should be coupled to funding. Software should be a science output in every discipline, as much as patents are in engineering. More recognition/citation counts for software. No more dissemination, researchers choose to hide progress/results, rather than 'publishing them'. Actual collaborative hunan machine intelligence, content mathematics finally machine readable, using open standards.

What are the problems, and are there solutions?

- Enriching the way research is attributed.
- Other things than papers being counted as research output.
- Will we use publications as a base to reward people/institutions?

- Strong support for open access.
- Do non-standard scientist gets skewed towards becoming more "normal"?
- Same people that were previously interested in interdisciplinarity now interested in software and data.
- There is no weird scientist, but there is an individuality and it should be accepted/rewarded as such.

Interplay between different parts of the publication ecosystem.

- Culture has to change in the top of universities, they should be happy to engage in different ways of research dissemination. These people also have the power to improve/change the REF structure.
- Nobody seems to like th REF, so why are we constructing/applying it?
- Funders also have considerable influence.

What further work could be done, and who should do it (make a pledge)?

Funders:

- Reward software output and public datasets
- Recognize the role of the Research Team, as opposed to individual excellence.

Publishers

- Innovation in submission/authorship process, encourage monetary discount for more complete and comprehensive submissions.
- Turn best practices into mandated customs.

Incorporate reproducibility into journal policies.

Credit reviewers, even after publishing, perhaps even publish the e-mail stream around the reviewers?

Academics:

- Improve curation of data, software and methods in ongoing academic work. Strive for reproducibility/repeatability.
- Participate in policy activities, e.g. guiding the next REF, and advicing on how to judge all academic outputs.
- Make software available that generates the research data and incorporate that in the publishing culture.

Are research software engineers doomed to obsoletion, and why?

What are the five most important things learnt during this discussion:

1. Researchers having a software development skill-set is a noble goal ... like world peace or global nuclear disarmament.
2. Some researchers will not have the interest, inclination, time to learn software development skills ("If I wanted to be a software developer then I'd have chosen that as a career path")

3. Concept of "division of labour" is sometimes forgotten.
4. RSEs understand the different drivers of researchers and developers and bridge the gap between the two, between exploring ideas, and implementing them as usable, maintainable, useful products.
5. Obsoletion is indeed a real word.

What are the problems, and are there solutions?

RSEs offer a pragmatic way to improve the state of software development within research and "A good plan ... executed now is better than a perfect plan executed next week."

Are there any useful resources that people should know about?

- https://www.software.ac.uk/blog/2012-08-16-what-research-software-community-and-why-should-you-care
- https://www.software.ac.uk/blog/2013-08-23-ten-reasons-be-research-software-engineer
- https://www.software.ac.uk/blog/2012-11-09-craftsperson-and-scholar

What is the minimum standard of quality for research software that we should expect?

General consensus of the conversation was that software should:

- Work (probably!) or at least describe the intended function of the tool (even it it doesn't work).
- Should be able to produce results of comparable quality?
- Should be able to replicate the described function of the software (through tests, sample data etc.)
- Share source code (not just binaries).

How we defined minimum:

- Minimum is: Make it clear what the licence is, you must put your name on it. The name is important because it gives you a contact, also a community identifier. If there is no author identifier then there is no identity behind the software.
- Licence and source code. Some software is shipped behind a web service and it's not possible to inspect source.
- Licence, Readme.
- Someway to identify function e.g. a command line with some sample data. Verify behaviour. Minimum standard should be access to the source code (especially for research software).
- Would just take a test suite over documentation.
- Level of documentation depends upon how complex the application/setup is. If the setup is hard then you need to verify each step. Code should follow conventions, Oxygen comments inline where necessary. Documentation should be tailored to the environment, conventions etc.

- Different levels of reproducibility, minimum standard depend upon the scientific domain and task.
- Versioned, test cases, releases

What programming languages, language features and paradigms support reproducibility?

What are the five most important things learnt during this discussion:

- Code needs to be comprehensible / easily-understandable. Do current languages do this? Probably not.
- Documentation is really important. GO-lang forces indention, does not compile if style/readability not upheld -> helps reproducibility-ish
- Code should be modular. Interface should be separate from implementation. Abstraction is the key. e.g. C++ header files tell you the interface. The 'class' system
- Data as code. Supply test data (non-binary) to validate the code
- Interoperability of languages.

What are the problems, and are there solutions?

Problems:

Object-Orientated interfaces don't capture many aspects of problems. e.g. Dimension types can be put in as comments, but that would be better encoded in the language (such as in F#). Trade-off between performance & reproducibility. Hard to fast AND readable.

Solutions:

- (relative to Problem #1) C++, Haskell, GO-lang ... F-sharp builds in....
- UNIT TESTING | UNIT TESTING | UNIT TESTING
- Make code modular
- Provide test data, and expected output
- Test system as part of the language, compiler runs these tests
- Automated ways to install the dependencies without VMs & hassle - if I can't install I can't reproduce
- Package Management as part of the compiler
- Versioning
- Must be open NOT proprietary, using open source licenses

What further work could be done, and who should do it (make a pledge)?

- Dominic pledges to do further researchers into this.
- Group: collaboration needed, work together

Are there any useful resources that people should know about?

- Yes. F-sharp has cool features and integrates with Azure
- GO-lang has some interesting features like an integrated package manager in the compiler

How should researchers be trained in producing reproducible research?

What are the five most important things learnt during this discussion:

- Software Carpentry integrated into Doctoral Training Programs
- Workflow-orientated tools
- Training for PI's, department heads, in appreciating the advantages in reproducible scientsts.
- Training for 'do-ers' (post-docs, PhD students) on how to do reproducible science
- Getting people to reproduce other work.

What are the problems, and are there solutions?

- Cultural changes – training and rewards/benefits
- Start training earlier, easier to make it the accepted way. Need to convince people who are further into their career.

What further work could be done, and who should do it (make a pledge)?

Get people to reproduce research as training courses.

Are there any useful resources that people should know about?

Software Carpentry, Galaxy, opensciencetraining.com, MOOCS.

What programming languages, language features and paradigms support reproducibility?

What are the five most important things learnt during this discussion:

- Community Infrastructure - forums, library availability, support
- Documentation - core APIs and libraries, doc generation and exposure
- Compartmentalised environments - virtualenv, JRE/JARs, ruby gem files, static linking
- Readability - reuse, trust and reproducibility are underpinned by readability
- Code convention - naming, module structure, automated builds, testing / TDD

What are the problems, and are there solutions?

Access to community, lack of direction/support for novices or researchers that are "left to their own devices",
Solutions yes! (see important things)

What further work could be done, and who should do it (make a pledge)?

- Promoting best practice - Software Carpentry
- Start learners on the road to best practice early

- Lobby PIs to recognise that courses and training on programming languages and best practice is a good use of time for a student

What are the best practices for domain-based and

institutional data management for supporting reproducible research?

What are the five most important things learnt during this discussion:

- Concise institutional strategies required, especially to engage culture change regarding RDM
- Digitisation of different forms of data (qualitative vs quantative, other formats) - is the focus on the OBJECTS or on the ATTRIBUTES of the objects
- Role of librarians/info specialists is important - e.g. embedding in research groups to understand data types and management approaches required, assisting with RDM strategies to comply with funder policy (e.g. EPSRC requirements by 2015)
- Effective training of researchers in RDM - e.g. Edinburgh's MANTRA project, covers various areas. E.g. organising data, licensing data, file formats and transformation
- Need to evaluate whether current method of recording sciences is sufficient to reproduce the science. E.g. over time in a lab, people usually become more similar can learn to do and see the same things. When transported to another lab, this is not necessarily the case, and can't count on it. To get the same reproducible research need same conditions, environment, data, expertise. Having the data alone is not enough.

What are the problems, and are there solutions?

- Universities are slow to sort RDM infrastructure - one example committee has been going for 10y and still not there
- Humanities don't have the same requirements for RDM as quant subjects
- Portability/accessibility - e.g. LOCKSS
- What further work could be done, and who should do it (make a pledge)?
- More metastudies into how research is done in the first place to ensure comprehensive enough knowledge in order to make it sustainable

Are there any useful resources that people should know about?

Mantra - RDM