CODECHECK: An open-science initiative to facilitate sharing of computer programs and results presented in scientific publications.

Stephen J Eglen (University of Cambridge) and *Daniel Nüst* (University of Münster).
Supported by Mozilla mini science award and UK Software Sustainability Institute.

WHAT IS IT?

1. CODECHECK is a system for independent verification of computations reported in publications.
2. A CODECHECKER is a reviewer of the code, checking that it works in an independent environment.
3. CODECHECKER writes a certificate showing what could be reproduced using the author's data and code.
4. This certificate increases trust in the results presented in scientific publications and allows others to reuse the code.

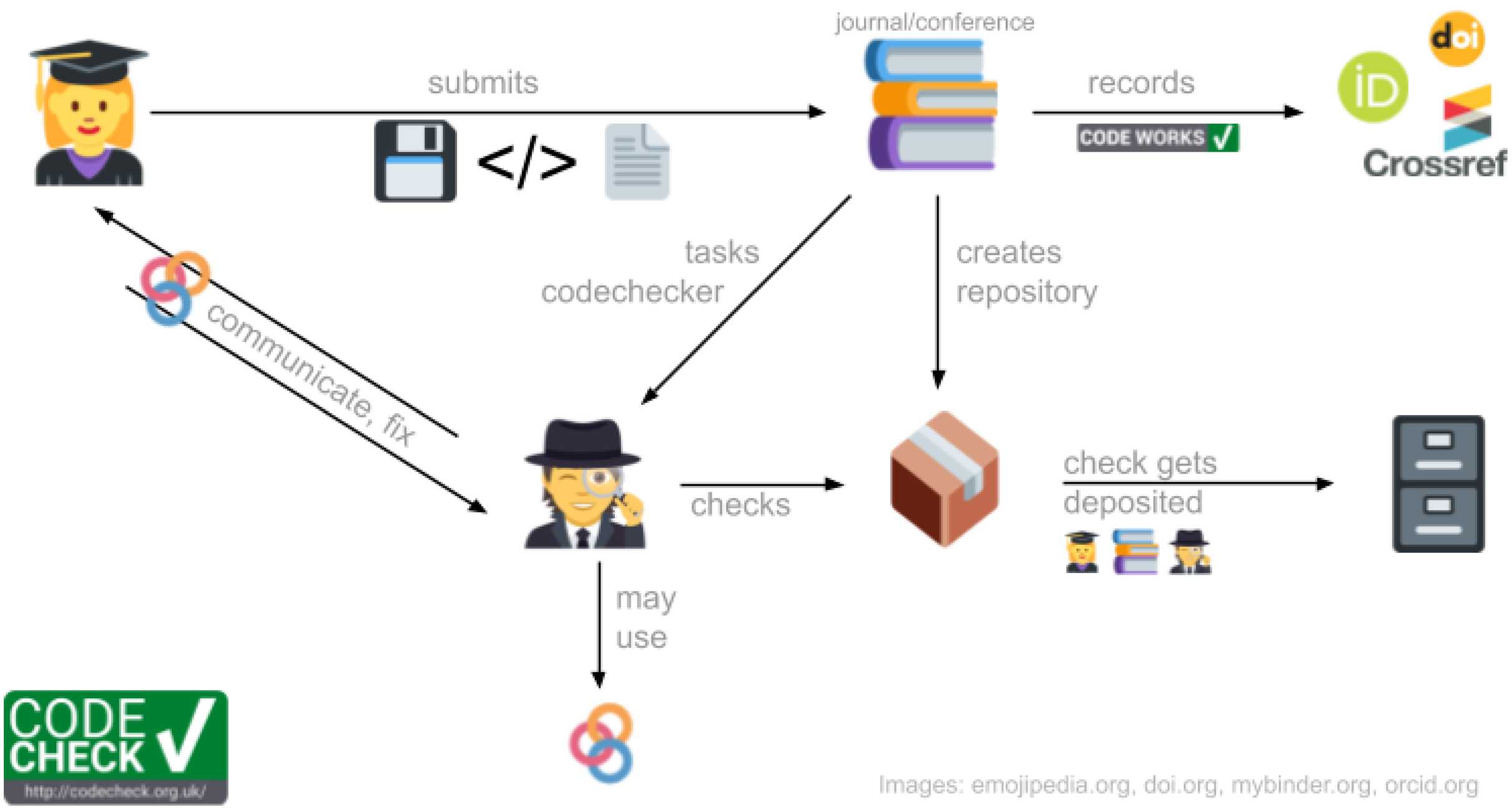
WHY?

- **Problem: Computer programs underlying publications are valuable artifacts, yet are currently rarely archived or shared.**
- Some solutions emerging, e.g. CODE OCEAN ("reproducible forever for everyone").
- We deliberately set the bar low: "reproducible once for someone else."
- We believe that sharing code, in whatever state, is better than nothing; having some of it independently checked is useful.

HOW DOES IT WORK?

1. AUTHOR submits code and data.
2. CODECHECKER runs code and data; iterates with AUTHOR until "code works".
3. CODECHECKER writes certificate stating what outputs were generated in their computing environment.
4. Certificate, with code and data, submitted to ZENODO.
5. Certificate is then shared by AUTHOR and PUBLISHER to the community. (Badge or URL in methods.)

HOW DOES IT WORK?



WHO DOES THE WORK?

AUTHOR must provide code and data, and assume it will be freely available at some point. Must also provide a description (either human or machine readable) description of how to run code.

CODECHECKER has to run code and check that it works, then write short certificate.

PUBLISHER can oversee this process, and refers to certificate in article.

WHO BENEFITS?

AUTHOR Can run check before submission and confirm "code works" for someone else. Archival for future reference.

PEER REVIEWERS If certificate available during peer review, reviewer can read it.

PUBLISHER Gets a code/data/results bundle to share.

READERS Can check certificate contents and immediately build upon the work.

PRINCIPLES

1. **CODECHECKERS ARE HUMANS AND COMMUNICATION IS KEY.** Codechecker will likely need to communicate with author to iterate on checking software.
2. **CODECHECKERS RECORD BUT DON'T INVESTIGATE OR FIX.** If any problems are found, the author is informed of the problem and normally expected to fix it.
3. **CREDIT IS GIVEN TO CODECHECKERS.** Certificate is authored by CODECHECKER and deposited in Zenodo. Metadata about certificate deposited to public databases (e.g. CrossRef, Publons).
4. **WORKFLOWS ARE SCRIPTED, AUDITABLE, AND THEY WORK.** Codechecker will write scripts (Makefile, Dockerfile) to automate the process and provide an audit log of what was done.

TECHNOLOGY

1. Author provides code + data. We need LICENSE, MANIFEST, and README or Makefile.
2. Codechecker repository on github stores code and data.
3. Use software (*Docker, python virtualenvs, R renv*) for fresh environments.
4. *Makefile* to provide human- and machine-readable description of workflow.
5. *mybinder* for interactively sharing environments.
6. *Rmarkdown* currently used to author certificates.
7. Certificates deposited on *Zenodo* by Codechecker.
8. Currently reliant on publishers for deposition of metadata to relevant sites (ORCID, CrossRef, Publons).

LIMITATIONS

1. Relies on a human Codechecker, so needs resource (and credit) to succeed.
2. Author's code and data must be freely available.
3. Easy to pass the CODECHECK (e.g. by embedding output figures in code!), but as code/data are open, should be easily found.
4. Threshold for certificate is low (minimum suggested: one figure from a paper).
5. High-performance compute examples may be too demanding to replicate.

WHAT NEXT?

1. Generating workflows consistent with principles that adapt to situation (particular needs for publishers, or author testing pre-submission).
2. Develop and train a pool of Codecheckers, perhaps Early Career Researchers.
3. Continue discussions with journals *eLife, Gigascience, Scientific Data*.
4. Generating portfolio of certificates in computational neuroscience to act as examplars.
5. Check our website for further information: http://codecheck.org.uk